located at each of the plurality of head-ends 210. In either embodiment 1 or 2, the interrelationships between the management tools, provider equipment and subscriber equipment is functionally the same.

Referring back to FIG.3, the remote head-end 110R is linked to the plurality of local head-ends 110 through an inter-server network 160 via switch 170. Switch 170 also interconnects the access network 150 and an inter-server network 160. As such, any communications between each of the video stream servers 114 and the stream session manager pass through the switch 170. Likewise, any communications between the content session manager 124 and the remote head-end 110R also pass through the switch 170.

Flow of video information from the remote head-end 110R may take either of two paths. Across the inter-server network 160 through the switch 170 and into the local video stream server 114 at the local head-end 110, or across the inter-server network 160 through the switch 170 and across the access network 150, directly to the subscriber equipment 140.

The content manager 120 also monitors the frequency of subscriber requests for video assets. This allows the interactive information distribution system 100 to control video asset storage and distribution, based on the frequency of requests when compared to a threshold rate. A threshold rate is a value for each requested video asset, established by the service provider in the interactive information distribution system 100, which defines a level to be considered as frequent or infrequent requests by the subscribers.

Each video asset may have multiple threshold rates. Multiple threshold rates are set to establish various parameters for the storage locations of video information. Such parameters include discarding the video asset, storing it at a single head-end 110, replicating the video asset and storing it at more than one head-end 110 where the request rate warrants it, or storing it at all the head-ends 110 across the entire interactive information distribution system 100.

Threshold rates may also be utilized to determine storage at a particular head-end based on the time of day when a video asset is most frequently requested. For example, video assets pertaining to children's shows are mostly viewed during the day before 6:00 p.m. In the evening, such video content may be discarded at a

majority of the head-ends so as to allow for the storage of adult type content usually viewed in the evening by the adult subscribers.

The content session manager 124 updates the content manager 120 of any changes in the distribution of video assets amongst the local video stream servers 114. The content manager 120 at each head-end 110 maintains files regarding the storage and distribution of various movies and other video assets across the entire interactive information distribution system 100.

In operation, at each of the local head-ends 110, the stream session manager 122 controls a navigation session by communicating with the subscriber equipment 140. The subscriber selects and requests a choice of video content, such as a movie, using a control device 146 of the subscriber equipment 140. The stream session manager 122 sends the request from a set-top box 142 in the subscriber equipment 140 to the content manager 120. The content manager 120 then queries its inventory files as to the location of the content.

If the content manager 120 determines that the requested video asset is located at the local head-ends 110 on the primary storage partition 118, then the stream and session manager 122 is notified to deliver the requested video asset. The stream session manager 122 instructs the video stream server 114 to transmit the video information through the access network 150 to the subscriber equipment 140 for viewing on the display 144.

If the content manager 120 determines that the requested video asset is not located at the local head-ends 110, but rather on the secondary storage location 119 at the remote server head-end 110R, the stream and session manager 122 contacts the content session manager 124 through the content manager 120. The content session manager 124 regulates transmission and flow of the requested video assets from the remote server 114R to the local video stream server 114.

The content session manager 124 then instructs the remote server 114R to transmit the video information through the inter-server network 160 to the requesting local head-end 110. Once received by the requesting local head-end 110, the video assets are transmitted over the access network 150 to the subscriber's equipment 140 for viewing on the display 144. Alternately, if a threshold value has been traversed, the local video server 114 is notified by the content manager 120 to

begin storing the video asset on its storage device 116 prior to its transmission over the access network 150 to the subscribers.

FIGS. 4A and 4B together depict a flow diagram representing a method 400 of allocating video information based upon subscriber requests. It further demonstrates the method of distributing video information. The method 400 is used in an interactive information distribution system 100 or 200 of FIG's 1 and 2. The subscriber selects a video asset from a title listing on the display of the subscriber's equipment. At step 404, a request is transmitted to the local head-end by the subscriber whereupon in step 406, a stream session manager and a content manager receive the request. In step 410, the content manager queries its files to determine if the requested video information is currently stored on the primary storage partition at the local head-end.

If the video information is locally stored on the primary storage partition at the local head-end, then the method 400 proceeds to step 412, where the content manager notifies the stream and session manager to transmit the requested video information. The stream and session manager controls the transmission and receiving of streamed video information on the local video stream server. In step 414, the stream and session manager notifies the video stream server to transmit the requested video information. In step 416, the video stream server transmits the video information, through the access network, to the subscriber's equipment for viewing on a display and the method ends at step 417.

If, in response to the query at step 410 that is negatively answered, where the video information is not locally stored on the primary storage partition at the local head-end, the method proceeds to step 418 (FIG. 4B) where the stream and session manager sends the request to the content session manager. In step 420, the content session manager then identifies the video information stored on the remote storage location (i.e., the secondary storage partition at the remote server head-end, as depicted by the first embodiment of this invention in FIG.1, or the secondary storage partition at each head-end, as depicted by the second embodiment of this invention in FIG.2).

After the video information is identified in step 420, the content session manager, at step 422, allocates the necessary bandwidth on the inter-server network. At the remote head-end, the content session manager determines the most efficient

route to allocate the video asset to the requesting head-end. The content manager allocates a path for transmission based upon factors such as network bandwidth, popularity of the requested asset in view of subscriber requests, and storage capacity at the local server.

At step 424, the method 400 queries whether the video assets are to be transferred to the local video server, as opposed to being streamed directly to the subscriber. If the query of step 424 is affirmatively answered, the method proceeds to step 426 where the content session manager notifies the remote server to begin streaming to the local server. The remote server will wait for a response from the local server signifying that the local server is ready to receive the video stream. If at step 428, no reply is given in some pre-designated time, the remote server will continue to notify the local server until a "ready" response is received. Once the remote server receives the ready signal, the method 400 proceeds to step 430 where the remote video server streams the video information over the inter-server network to the local video server. The local video server then transmits the video information to the requesting subscriber. If at step 424 the query is negatively answered, i.e., the method determines that the video information at the remote head-end is not to be transferred to a local video server, then at step 432, the content session manager additionally requests the stream session manager to allocate bandwidth through the access network.

Once the bandwidth on the access network is allocated, the content session manager notifies, in step 434, the stream session manager at the remote head-end to start streaming the requested video asset and at step 436, the remote stream session manager then signals the remote stream server to begin streaming the video. In step 438, the remote stream server transmits the video asset over the allocated bandwidth of the inter-server network, across a network interface switch, and through the access network to the final destination, the subscriber equipment. The method 400 ends at step 439. These alternate methods provide effective options to transmit video assets under various conditions in the interactive information distribution system.

FIG.5 depicts a flow diagram representing a method 500 of storing video assets at the various locations in the interactive information distribution system. The

method is used in an interactive information distribution system such as an interactive video distribution system 100 or 200 of FIGS. 1 and 2.

The method 500 starts at step 501 and proceeds to step 502 where a plurality of subscribers transmit requests for video content from each of a plurality of subscriber equipment. In step 504, each local video stream server at a plurality of head-ends designated to support the plurality of subscribers, receives the subscriber requests and notifies the content manager thereof. At step 506, the content manager at each local head-end collectively quantifies a subscriber asset request rate for each video asset.

The service provider, in step 508, sets at least one threshold rate for each video asset in order to establish a frequency request level for subscriber requests of video assets. Then, at step 510, the subscriber asset request rate is compared to the threshold rate for each video asset.

The method 500 queries at step 512 whether the asset request rate traverses the threshold rate. If the query is affirmatively answered, then, in step 514, the video asset is directed by the management tools to be streamed to the primary storage partitions at each of the plurality of head-ends designated to store the replicates of that video asset. The head-ends that store the replicated video asset are those based upon traversing the threshold request rate for that specific head-end. If one of the threshold rates for the entire system is traversed, then, at step 516, the video asset is stored amongst all of the plurality of head-ends in the interactive information distribution system.

Alternatively, if at the query of step 512 is negatively answered, the method 500 deems that the asset request rate does not traverse the threshold rate. At step 518, the video asset is directed by the management tools to be stored at the secondary storage partition. In step 520, some or all of the video assets that are replicated on the primary storage partitions may be removed, depending on the subscriber request rate at that specific head-end, and the method 500 ends at step 521.

In the first embodiment as depicted in FIG.1, the infrequently requested video assets are stored on the secondary storage partition 119 at the remote server head-end 110R. In the second embodiment as depicted in FIG.2, the infrequently requested video assets are divided amongst each of the plurality of head-ends and then stored on their respective secondary storage partitions. In either the first or

second embodiments, there is a separation of frequently requested video assets from infrequently requested assets. The result of this method is a reduction of storage capacity required for storing the thousands of video assets in duplication, while still providing the assets to the plurality of subscribers seamlessly without delay.

Another feature of this inventive system is file transfer to the subscriber at a faster rate than the subscriber equipment can retrieve the file. The video assets comprise a number of files that include the play track, fast-forward track, rewind track, movie information screen (MIS), preview, etc. When the file is transferred from the video stream server, multiple file transfers may be launched simultaneously faster than real time. For example, normal play track can be transferred at nine times real time (e.g. 30 MB/sec), fast-forward track at real time (3.3 MB/sec), and rewind track in reverse in real time.

These files may be simultaneously transferred by the video stream server. They are transmitted by the server (e.g., bursts) to the subscriber equipment at a faster rate than the subscriber can retrieve these files, even at the fast-forward rate of transfer. For example, if the video server transmits a movie to a subscriber, who, at the same time plays the movie in the fast-forward mode, the video stream server will transmit the movie before the subscriber will finish playing the movie in fast-forward mode. In this way, the stream server will satisfy a new customer request for the title, even though the title has not been completely transferred from the remote server. Thus, as long as the transfer from the stream servers is faster, it will always be ahead of the subscriber's access to the title.

It should be apparent to those skilled in the art that a novel system for storing and distributing video content on a plurality of servers within an interactive information distribution system has been provided. The methods and apparatus described herein eliminate the need for independent storage devices at each neighborhood site and redistributes much of the video content across the entire network of servers. Although various embodiments that incorporate the teachings of the present invention have been shown and described in detail herein, those skilled in the art can readily devise many other varied embodiments that still incorporate these teachings.

What is claimed is:

1. In an interactive information distribution system (200) including a network of provider equipment and subscriber equipment, apparatus comprising:

a plurality of servers (214) coupled to respective subscriber equipment (240), each of said servers (214) having a primary storage partition (218) for storing a local portion of video assets, each of said servers (214) having a secondary storage partition (219) for storing at least some of a remaining portion of said video assets; and

a manager (220, 222, 224), coupled to each of said plurality of servers (214) for routing video assets between said servers (214) in response to video asset requests, and for migrating video assets between storage partitions (218, 219) in response to a video asset request rate traversing a threshold rate.

2. The apparatus of claim 1, wherein:

said manager (220, 222, 224) allocates said video assets to at least one of said plurality of servers (214) for storage on said primary storage partitions (218) when said asset request rate traverses said threshold rate; and

said manager (220, 222, 224) stores said video assets on said secondary storage partition (219) when said asset request rate does not traverse said threshold rate.

3. The apparatus of claim 2, wherein:

in response to an asset request from subscriber equipment (240), said manager (220, 222, 224) distributes to said requesting subscriber equipment (240) the requested video asset from a server (214) storing the requested video asset.

4. The apparatus of claim 1, wherein said manager comprises:

a content manager (220), coupled to said plurality of servers (214) for tracking, inventorying and administering said asset request rate and said threshold rate for each of said video assets;

a stream session manager (222), coupled to said plurality of servers (214) and linked with said content manager (200), for distributing streams of video assets to subscriber equipment (240) requesting said video assets;

a content session manager (224), coupled to said content manager (220) and said plurality of servers (214), for receiving asset requests from said stream session manager (222) via said content manager (220); and

an access network (250), coupled between each of said plurality of servers (214) and said respective subscriber equipment (240), for receiving asset requests and transmitting video assets.

5. The apparatus of claim 4, wherein a server (214), identified by said content manager (220) as storing a requested video asset, provides said requested video asset to requesting subscriber equipment (240) via said access network (250).

6. The apparatus of claim 5, wherein said video asset is stored on said primary storage partition (218) or secondary storage partition (219) of at least one of said plurality of said servers (214) correspondingly linked to said subscriber equipment (240).

7. In an interactive information distribution system (100) including a network of provider equipment and subscriber equipment, apparatus comprising:

a plurality of local servers (114) having a primary storage partition (118);

a remote server (114R) having a primary storage partition (118R) and a secondary storage partition (119R); and

a manager (120, 122, 124), coupled to each of said plurality of local servers (114) and said remote server (114R), for routing video assets between said remote server (114R), said plurality of servers (114) and said subscriber equipment (140) in response to video asset requests, and for migrating video assets between storage partitions (118, 118R, and 119R) in response to a video asset request rate traversing a threshold rate.

8. The apparatus of claim 7, wherein

said manager (120, 122, 124) allocates said video assets to at least one of said plurality of local servers (114) for storage on said primary storage partitions (118) when said asset request rate exceeds said threshold rate; and

said manager (120, 122, 124) stores said video assets on said secondary storage partition (119R) of said remote server (114R) when said asset request rate does not exceed said threshold rate.

9. The apparatus of claim 8, wherein:

in response to an asset request from subscriber equipment (140), said manager (120, 122, 124) distributes to said requesting subscriber equipment (140) the requested video asset from a local server (114) storing the requested video asset.

10. The apparatus of claim 9, wherein said manager comprises:

a content manager (120), coupled to said plurality of local servers (114) and said remote server (114R) for tracking, inventorying and administering said asset request rate and said threshold rate for each of said video assets;

a stream session manager (122), coupled to said plurality of local servers (114) and said remote server (114R), and linked with said content manager (120), for distributing streams of video assets to subscriber equipment (140) requesting said video assets;

a content session manager (124), coupled to said content manager (120) and said remote server (114R), for receiving asset requests from said stream session manager (122) via said content manager (120); and

an access network (150) coupled between each of said plurality of local servers (114) and said respective subscriber equipment (140) for receiving asset requests and transmitting video assets.

11. The apparatus of claim 10, wherein said local server (114) transmits said video asset across said access network (150) to said subscriber equipment (140).

12. The apparatus of claim 11, wherein said video asset is stored on said primary storage partition (118) of at least one of said plurality of said servers (114) correspondingly linked to said subscriber equipment (140).

13. In an interactive information distribution system (500) comprising a plurality of servers coupled to respective subscriber equipment, each of said servers having a primary storage partition for storing a first portion of video assets and a secondary storage partition for storing at least some of a remaining portion of said video assets, said servers providing video assets to respective subscriber equipment in response to subscriber requests, a method comprising the steps of:

determining (506) an asset request rate for each of said video assets stored in each server;

comparing (512) said determined asset request rates with respective threshold rates; and

in the case of (512) video assets stored on a secondary partition having a request rate exceeding said respective threshold rate, migrating (514, 516) said video assets stored on said secondary storage partition to a corresponding primary storage partition.

14. The method of claim 13, wherein in the case (512) of said determined asset request rate for video assets stored in a primary storage partition being below a respective threshold rate, migrating (518) said video assets from said primary partition to a corresponding secondary partition.

15. The method of claim 14, further comprising the steps of:

identifying (408) a server having a primary partition storing a requested video asset;

causing (414, 426) said identified server to begin providing said requested video asset; and

transmitting (416, 430) said video asset through an access network to said subscriber equipment initiating said asset request.

16. The method of claim 15, wherein (412) said identified server is coupled directly to said requesting subscriber equipment.

17. The method of claim 15, wherein said (418) identified server is coupled to said requesting subscriber via an intervening server, said identified server communicating (426) with said intervening server via an inter-server network.

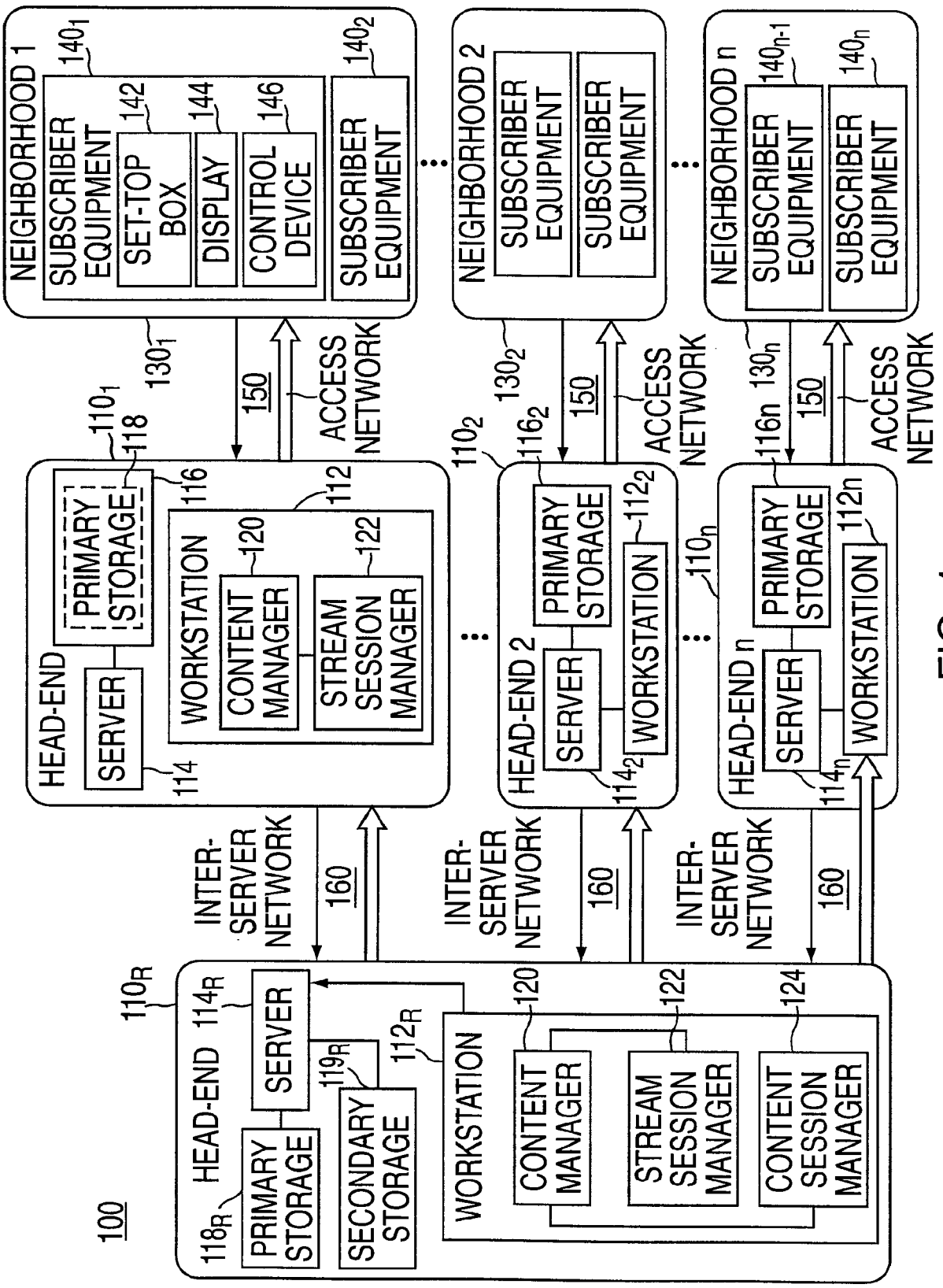

100
NEIGHBORHOOD 1
SUBSCRIBER EQUIPMENT
SET-TOP BOX
DISPLAY
CONTROL DEVICE
SUBSCRIBER EQUIPMENT
NEIGHBORHOOD 2
SUBSCRIBER EQUIPMENT
SUBSCRIBER EQUIPMENT
NEIGHBORHOOD n
SUBSCRIBER EQUIPMENT
SUBSCRIBER EQUIPMENT
ACCESS NETWORK
150
HEAD-END
PRIMARY STORAGE
SERVER
WORKSTATION
CONTENT MANAGER
STREAM SESSION MANAGER
HEAD-END 2
HEAD-END n
INTER-SERVER NETWORK
160
HEAD-END
SECONDARY STORAGE
CONTENT SESSION MANAGER

FIG. 1

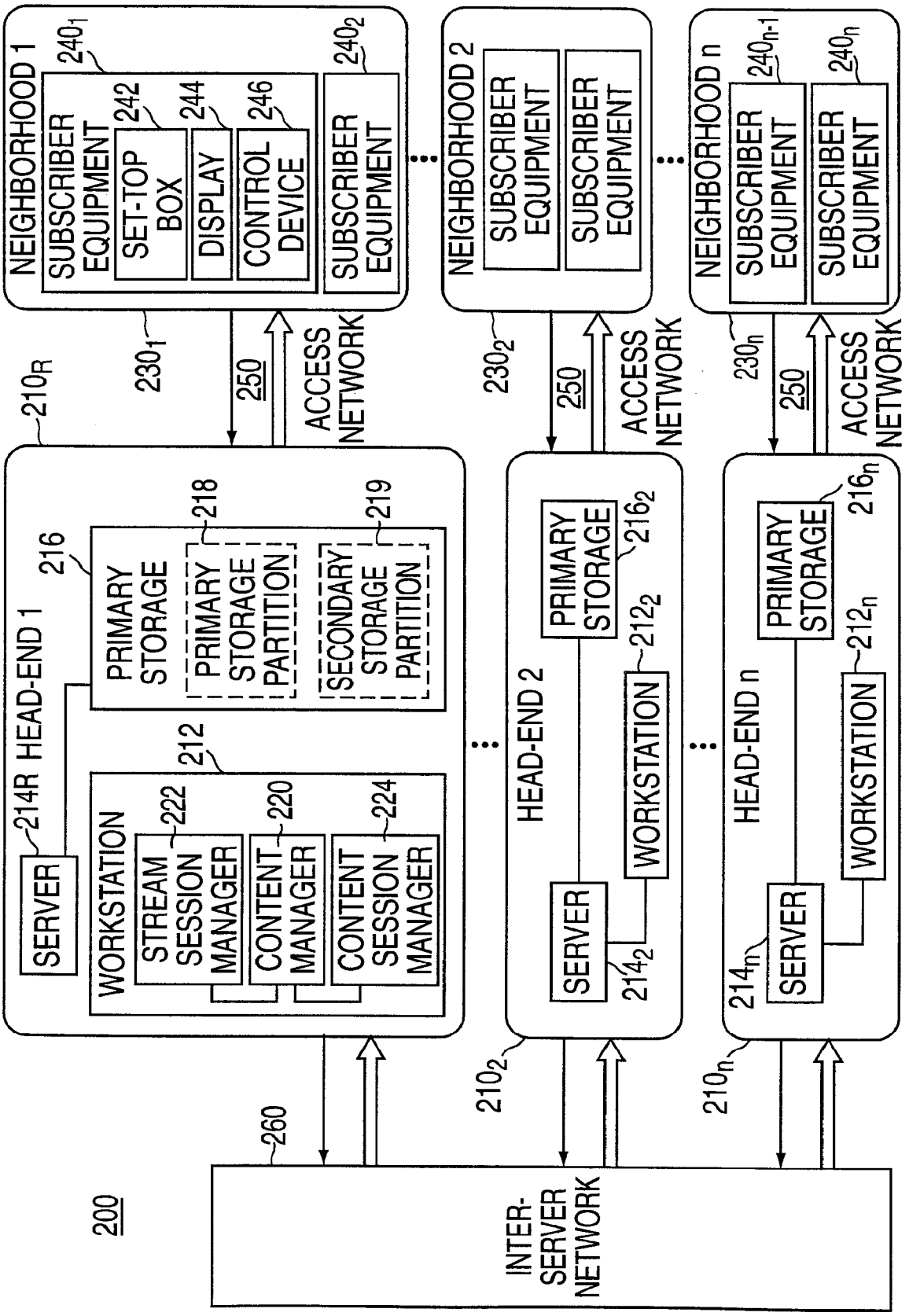

NEIGHBORHOOD 1
SUBSCRIBER EQUIPMENT
SET-TOP BOX
DISPLAY
CONTROL DEVICE
SUBSCRIBER EQUIPMENT
NEIGHBORHOOD 2
SUBSCRIBER EQUIPMENT
SUBSCRIBER EQUIPMENT
NEIGHBORHOOD n
SUBSCRIBER EQUIPMENT
SUBSCRIBER EQUIPMENT
ACCESS NETWORK
250
HEAD-END 1
HEAD-END 2
HEAD-END n
PRIMARY STORAGE
PRIMARY STORAGE PARTITION
SECONDARY STORAGE PARTITION
WORKSTATION
STREAM SESSION MANAGER
CONTENT MANAGER
CONTENT SESSION MANAGER
SERVER
INTER-SERVER NETWORK
200
260

FIG. 2

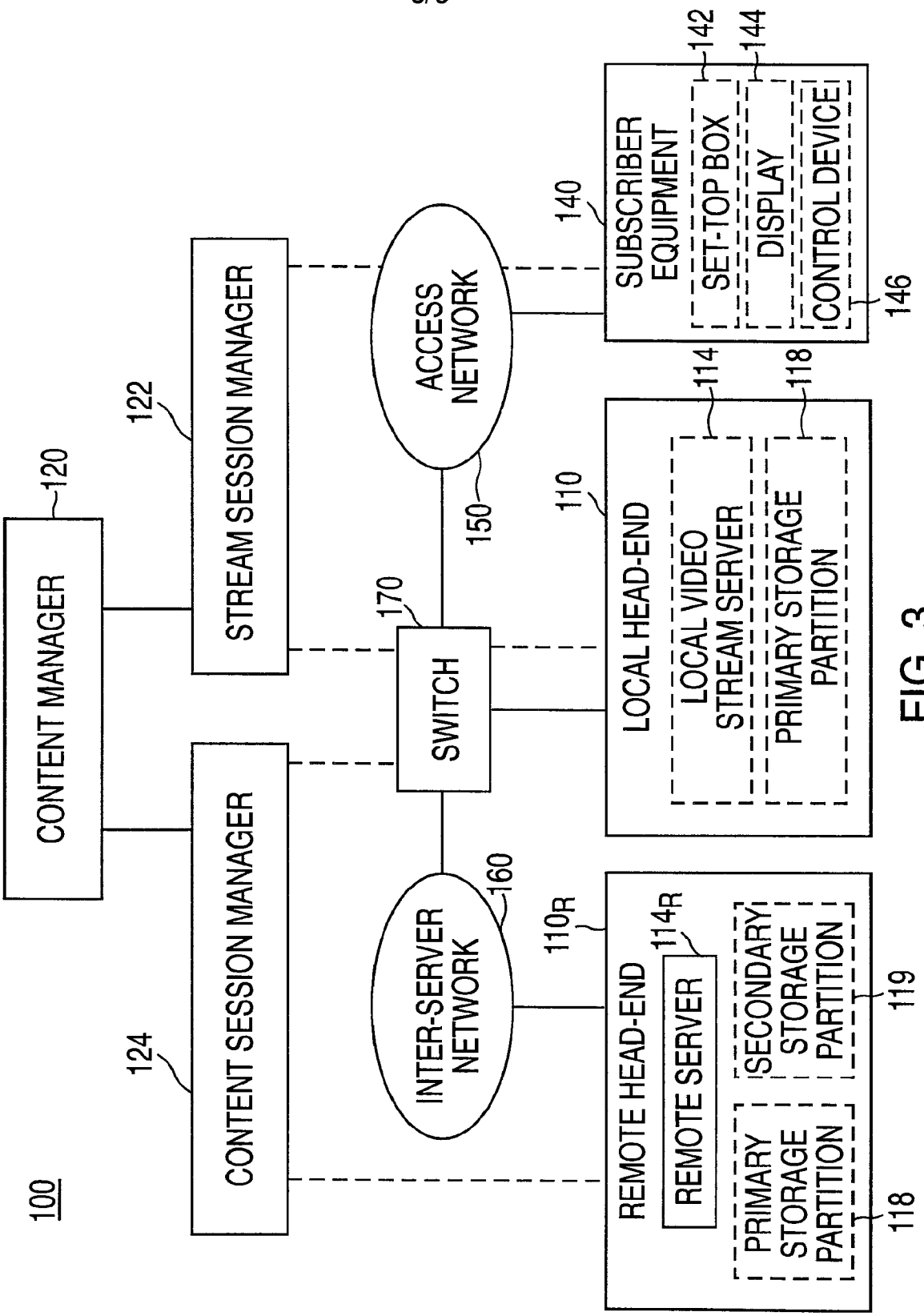

100
CONTENT MANAGER
120
STREAM SESSION MANAGER
122
CONTENT SESSION MANAGER
124
ACCESS NETWORK
150
SWITCH
170
INTER-SERVER NETWORK
160
SUBSCRIBER EQUIPMENT
140
SET-TOP BOX
142
DISPLAY
144
CONTROL DEVICE
146
LOCAL HEAD-END
110
LOCAL VIDEO STREAM SERVER
114
PRIMARY STORAGE PARTITION
118
REMOTE HEAD-END
110R
REMOTE SERVER
114R
PRIMARY STORAGE PARTITION
118
SECONDARY STORAGE PARTITION
119

FIG. 3

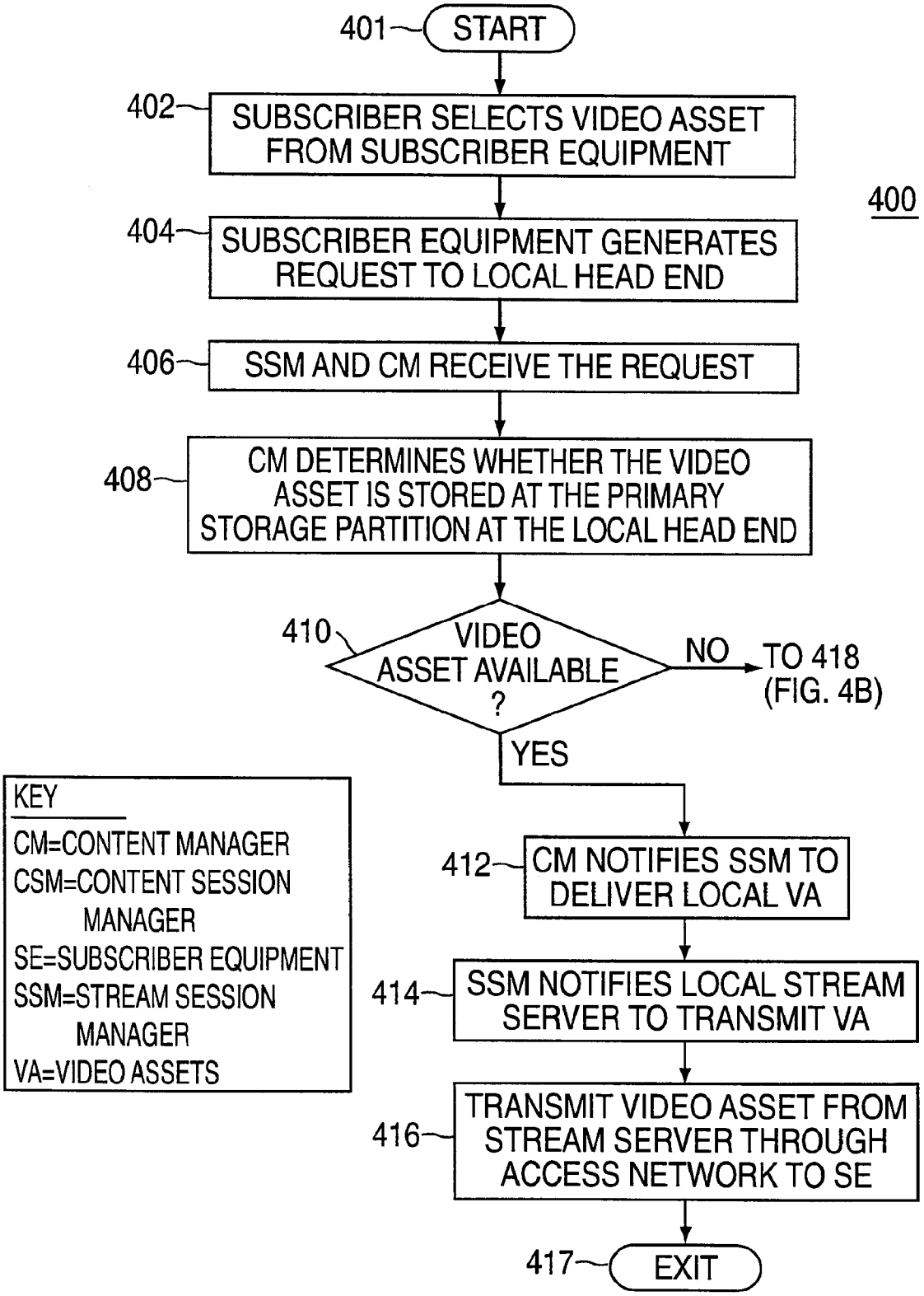

401
START
402
SUBSCRIBER SELECTS VIDEO ASSET FROM SUBSCRIBER EQUIPMENT
400
404
SUBSCRIBER EQUIPMENT GENERATES REQUEST TO LOCAL HEAD END
406
SSM AND CM RECEIVE THE REQUEST
408
CM DETERMINES WHETHER THE VIDEO ASSET IS STORED AT THE PRIMARY STORAGE PARTITION AT THE LOCAL HEAD END
410
VIDEO ASSET AVAILABLE ?
NO
TO 418 (FIG. 4B)
YES
KEY
CM=CONTENT MANAGER
CSM=CONTENT SESSION MANAGER
SE=SUBSCRIBER EQUIPMENT
SSM=STREAM SESSION MANAGER
VA=VIDEO ASSETS
412
CM NOTIFIES SSM TO DELIVER LOCAL VA
414
SSM NOTIFIES LOCAL STREAM SERVER TO TRANSMIT VA
416
TRANSMIT VIDEO ASSET FROM STREAM SERVER THROUGH ACCESS NETWORK TO SE
417
EXIT

FIG. 4A

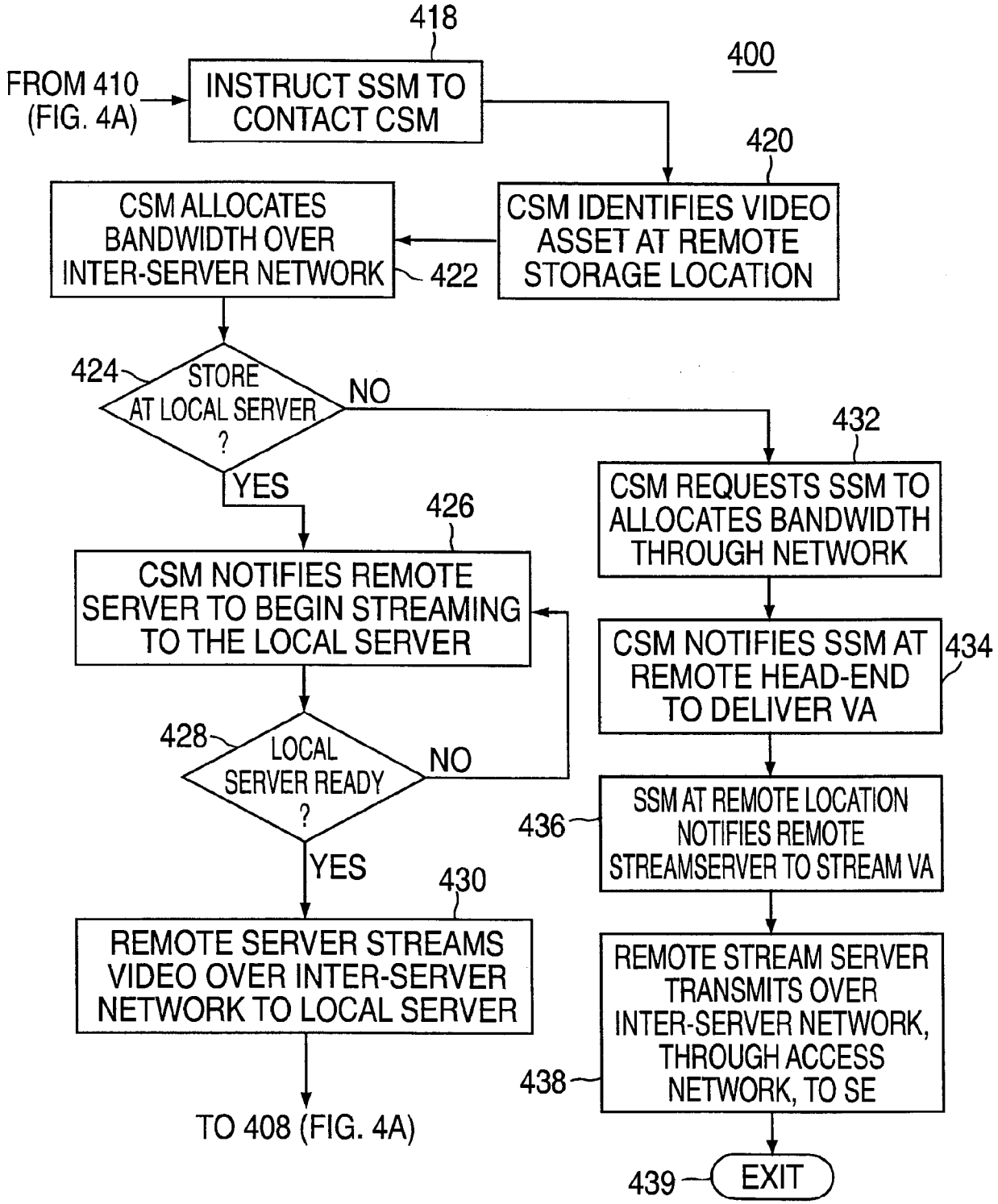

418
400
FROM 410 (FIG. 4A)
INSTRUCT SSM TO CONTACT CSM
420
CSM ALLOCATES BANDWIDTH OVER INTER-SERVER NETWORK
422
CSM IDENTIFIES VIDEO ASSET AT REMOTE STORAGE LOCATION
424
STORE AT LOCAL SERVER ?
NO
YES
432
CSM REQUESTS SSM TO ALLOCATES BANDWIDTH THROUGH NETWORK
426
CSM NOTIFIES REMOTE SERVER TO BEGIN STREAMING TO THE LOCAL SERVER
CSM NOTIFIES SSM AT REMOTE HEAD-END TO DELIVER VA
434
428
LOCAL SERVER READY ?
NO
YES
436
SSM AT REMOTE LOCATION NOTIFIES REMOTE STREAMSERVER TO STREAM VA
430
REMOTE SERVER STREAMS VIDEO OVER INTER-SERVER NETWORK TO LOCAL SERVER
REMOTE STREAM SERVER TRANSMITS OVER INTER-SERVER NETWORK, THROUGH ACCESS NETWORK, TO SE
438
TO 408 (FIG. 4A)
439
EXIT

FIG. 4B

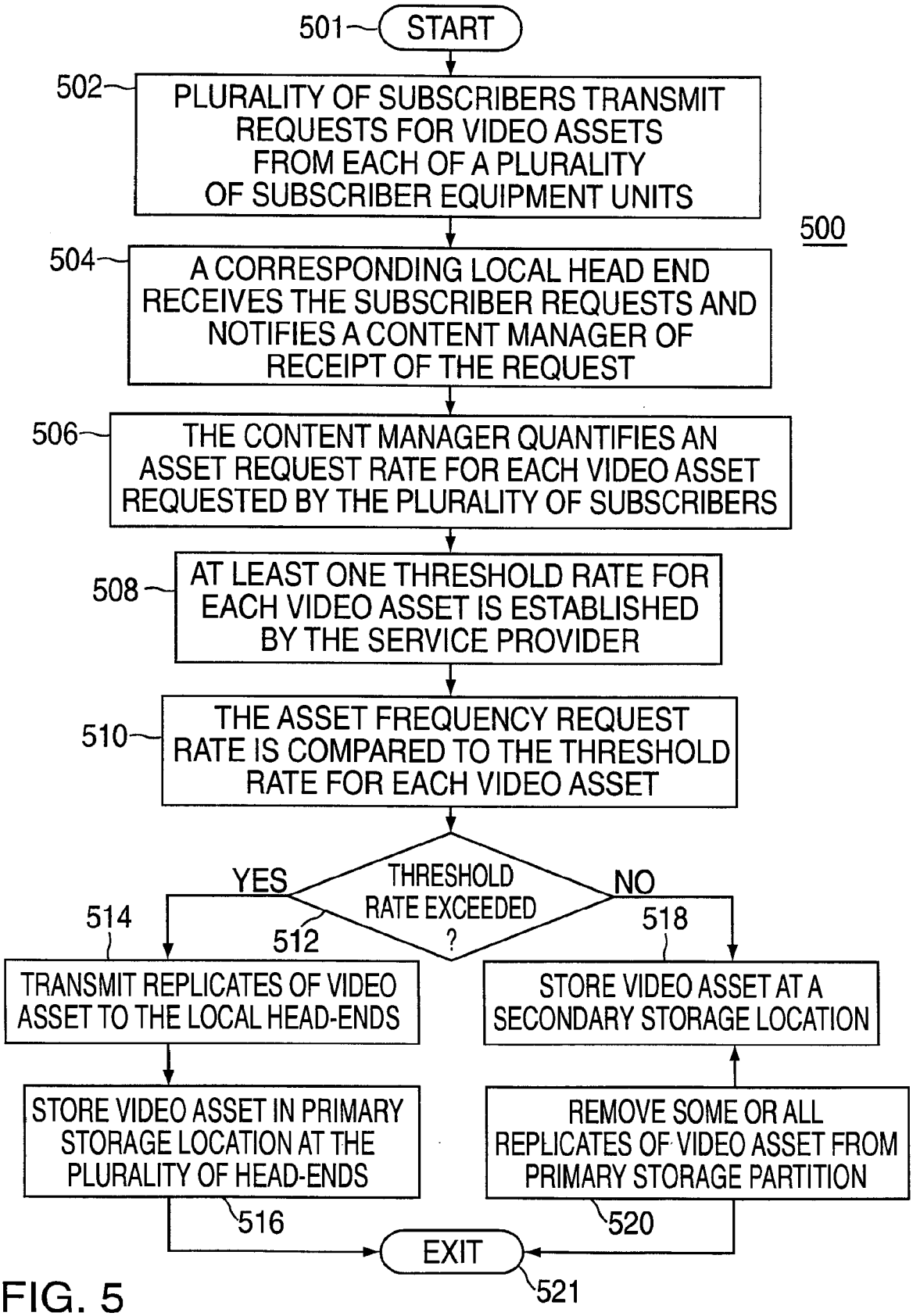

501
START
502
PLURALITY OF SUBSCRIBERS TRANSMIT REQUESTS FOR VIDEO ASSETS FROM EACH OF A PLURALITY OF SUBSCRIBER EQUIPMENT UNITS
500
504
A CORRESPONDING LOCAL HEAD END RECEIVES THE SUBSCRIBER REQUESTS AND NOTIFIES A CONTENT MANAGER OF RECEIPT OF THE REQUEST
506
THE CONTENT MANAGER QUANTIFIES AN ASSET REQUEST RATE FOR EACH VIDEO ASSET REQUESTED BY THE PLURALITY OF SUBSCRIBERS
508
AT LEAST ONE THRESHOLD RATE FOR EACH VIDEO ASSET IS ESTABLISHED BY THE SERVICE PROVIDER
510
THE ASSET FREQUENCY REQUEST RATE IS COMPARED TO THE THRESHOLD RATE FOR EACH VIDEO ASSET
THRESHOLD RATE EXCEEDED ?
512
YES
NO
514
518
TRANSMIT REPLICATES OF VIDEO ASSET TO THE LOCAL HEAD-ENDS
STORE VIDEO ASSET AT A SECONDARY STORAGE LOCATION
STORE VIDEO ASSET IN PRIMARY STORAGE LOCATION AT THE PLURALITY OF HEAD-ENDS
REMOVE SOME OR ALL REPLICATES OF VIDEO ASSET FROM PRIMARY STORAGE PARTITION
516
520
EXIT
521

FIG. 5

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US00/08402

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) :H04N 7/10
US CL :709/217, 219; 348/7
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 709/217, 218, 219; 348/6-13

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EAST

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 5,550,577 A (VERBIEST ET AL.) 27 AUGUST 1996, FIGURES 1, 3, 4 AND 5.	1-17
A,P	US 5,898,456 A (WAHL) 27 APRIL 1999, FIGURES 1-8.	1-17
A	US 5,768,681 A (DAN ET AL.) 16 JUNE 1998, FIGURE 1.	1-17
A	US 5,581,479 A (MCLAUGHLIN ET AL.) 03 DECEMBER 1996, FIGURES 2, 1-13.	1-17

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search	Date of mailing of the international search report
27 JUNE 2000	24 AUG 2000
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer NATHAN J. FLYNN Telephone No. (703) 308-0000

Eugenia Zoga

Form PCT/ISA/210 (second sheet) (July 1998)★

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2000－207270
（P2000－207270A）
(43)公開日　平成12年７月28日(2000.7.28)

(51)Int.Cl.[7]		識別記号	FI			テーマコード(参考)
Ｇ０６Ｆ	12/00	５４６	Ｃ０６Ｆ	12/00	５４６Ｋ	５Ｂ０８２
	13/00	３５４		13/00	３５４Ｄ	５Ｂ０８９

審査請求　未請求　請求項の数４　ＯＬ　（全 19 頁）

(21)出願番号　特願平11－6072

(22)出願日　平成11年１月13日(1999.1.13)

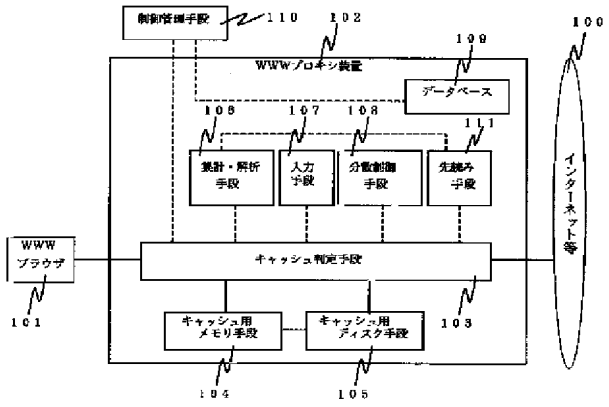
(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地
(72)発明者　安齋　友矢
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内
(72)発明者　吉田　健一
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内
(74)代理人　100068504
弁理士　小川　勝男

Ｆターム(参考)　5B082 AA00 FA12 HA02 HA08
5B089 GA19 GB02 HA01 JA21 KA01
KA06 KD02 KD05 LB04

(54)【発明の名称】　WWWプロキシ装置

(57)【要約】
【課題】WWWプロキシ装置において、高速で効率的、且つ信頼性の高いキャッシュ処理を行い、また、ユーザがリクエストをする可能性の高いWWWデータを先読み処理することにより、より効率的なキャッシュ処理、及びネットワークの負荷の低減を実現する。
【解決手段】キャッシュ処理にメモリ装置とディスク装置を使用し、通常はWWWデータをメモリ装置にキャッシュ処理し、キャッシュデータの置換により溢れたWWWデータについて、一定条件を満たすWWWデータのみディスク装置にキャッシュ処理することによりユーザのリクエストに高速にリプライすることができる。先読みにおいて、一定条件を満たすWWWデータを先読みの処理を行い、ネットワークの負荷を低減しキャッシュのヒット率を向上させる。

図１



【特許請求の範囲】
【請求項１】ＷＷＷデータをＷＷＷブラウザからの依頼でＷＷＷサーバ装置から代理取得するＷＷＷプロキシ装置において、
該ＷＷＷプロキシ装置が、ＷＷＷデータを一時的に記憶するためのキャッシュ用メモリ手段及びキャッシュ用ディスク手段と、
ネットワーク上のＷＷＷデータの転送のログ・データを集計・解析する集計・解析手段と、
ユーザからの指示を入力する入力手段と、
該入力手段及び該集計・解析手段の内容を基に該ＷＷＷデータを該キャッシュ用メモリ手段又は該キャッシュ用ディスク手段により記憶するか否かを判定し、該ＷＷＷデータの該キャッシュ用メモリ手段又は該キャッシュ用ディスク手段による書き込み及び読み出し、又は上位ＷＷＷプロキシ装置又はＷＷＷサーバ装置から該ＷＷＷデータの取得を行うキャッシュ判定手段とを具備し、
該キャッシュ判定手段が、通常は記憶すべき該ＷＷＷデータを該キャッシュ用メモリ手段にキャッシュし、該キャッシュ用メモリ手段のキャッシュデータの置換により溢れたＷＷＷデータについて、一定条件を満たすＷＷＷデータのみ該キャッシュ用ディスク手段にキャッシュすることを特徴とするＷＷＷプロキシ装置。
【請求項２】請求項１記載のＷＷＷプロキシ装置において、
該ＷＷＷプロキシ装置が、該集計・解析手段により集計されたログ・データのうち、最近の一定期間のログ・データのみのアクセス頻度の上位のＷＷＷデータのみキャッシュ処理を行うことを特徴とするＷＷＷプロキシ装置。
【請求項３】請求項１又は２記載のＷＷＷプロキシ装置において、
該キャッシュ用メモリ手段のキャッシュデータの置換により溢れたＷＷＷデータを該キャッシュ用ディスク手段にキャッシュするか否かを判定する上記一定条件が、ネットワーク上のＷＷＷデータの転送のログ・データを集計・解析する集計・解析手段により集計された該ＷＷＷデータのアクセス頻度の上位であることを特徴とするＷＷＷプロキシ装置。
【請求項４】請求項１又は２記載のＷＷＷプロキシ装置において、
該キャッシュ用メモリ手段のキャッシュデータの置換により溢れたＷＷＷデータを該キャッシュ用ディスク手段にキャッシュするか否かを判定する上記一定条件が、ユーザからの指示を入力する該入力手段の入力内容であることを特徴とするＷＷＷプロキシ装置。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、ＷＷＷ（World Wide Web）上において、ユーザのリクエストを受けて目的のサーバのＷＷＷデータの取得を代理で行い、取得したＷＷＷデータのキャッシュ処理を行うＷＷＷプロキシ装置に関する。
【０００２】
【従来の技術】ＷＷＷプロキシ装置を用いたＷＷＷネットワークシステムは、一般に図１４のように構成され、ユーザ側のＷＷＷクライアント１４０１からのリクエストを受けたＷＷＷプロキシ１４０２は、リクエストされたＷＷＷデータを目的のＷＷＷ１４０５サーバから代理で取得し、キャッシュ装置１４０３に記憶しておき、以後、キャッシュ装置１４０３に記憶してあるＷＷＷデータがユーザ側のクライアント１４０１によってリクエストされた場合は、キャッシュ装置１４０３内のＷＷＷデータをユーザ側のＷＷＷクライアントに返信する。
【０００３】この際、通常はキャッシュ装置としてディスク装置を用いるが、"Norifumi Nishikawa, Takafumi Hosokawa, Yasuhide Mori, Kenichi Yoshida, Hiroshi Tsuji, Memory-based architecture for distributed WWW caching proxy, ComputerNetworks And ISDN Systems (30)1-7 (1998) pp.205-214"に述べられているように、キャッシュ装置としてメモリ装置を用いてＷＷＷデータの書き込み、読み出しを高速化しようとする試みがある。
【０００４】また、ユーザからリクエストされたＷＷＷデータの参照先のＷＷＷデータをユーザがリクエストを行う前に取得しておく先読み方法については、参照されているＷＷＷデータの全てを取得し、キャッシュに格納しておく。ユーザが参照先のデータのリクエストを行えば、直ちにキャッシュに格納しておいたＷＷＷデータを返信することにより、ユーザ側の参照に要する時間が大幅に短縮される。
【０００５】
【発明が解決しようとする課題】上記従来の技術のＷＷＷプロキシ装置において、キャッシュ装置としてディスク装置のみ用いる場合に関しては、ＷＷＷデータのキャッシュ装置への書き込み及び読み出しの速度が低速になるために、キャッシュ処理及びユーザへのＷＷＷデータの送信に時間がかかるという問題がある。
【０００６】また、キャッシュ装置としてメモリ装置のみ用いる場合に関しては、キャッシュ装置への書き込み及び読み出しの速度は高速になる反面、メモリ自体がディスクに比べて高価であるため、大容量のキャッシュ装置として使用しづらい、障害時にキャッシュ内のＷＷＷデータが消失してしまうといった問題がある。
【０００７】また、先読み方法に関しては、リクエストされたＷＷＷデータの参照先の全てのＷＷＷデータの先読み、又はリクエストされたＷＷＷデータの参照先の全てのＷＷＷデータがさらに参照しているＷＷＷデータの先読みも行うために、ユーザが引き続いてリクエストを行う可能性の低いＷＷＷデータも多く先読みしてしま

い、結果としてネットワークの負荷が増大するという問題がある。

【０００８】

【課題を解決するための手段】上記従来技術の課題を解決するために、本発明におけるＷＷＷプロキシ装置は、ＷＷＷデータを一時的に記憶するためのキャッシュ用メモリ手段及びキャッシュ用ディスク手段と、ネットワーク上のＷＷＷデータの転送のログ・データを集計・解析する集計・解析手段と、ユーザからの指示を入力する入力手段と、該入力手段及び該集計・解析手段の内容を基に該ＷＷＷデータを該キャッシュ用メモリ手段又は該キャッシュ用ディスク手段により記憶するか否かを判定し、該ＷＷＷデータの該キャッシュ用メモリ手段又は該キャッシュ用ディスク手段による書き込み及び読み出し、又は上位ＷＷＷプロキシ装置又はＷＷＷサーバ装置から該ＷＷＷデータの取得を行うキャッシュ判定手段と、ＷＷＷプロキシ装置を制御管理するためのデータベースと、該データベースを用いて該ＷＷＷプロキシ装置を制御管理する制御管理手段と、リクエストされたＷＷＷデータの参照先のＷＷＷデータの先読みを行う先読み手段とを具備することを特徴とする。

【０００９】すなわち、本発明は、ＷＷＷデータをＷＷＷブラウザからの依頼でＷＷＷサーバ装置から代理取得するＷＷＷプロキシ装置において、該ＷＷＷプロキシ装置が、ＷＷＷデータを一時的に記憶するためのキャッシュ用メモリ手段及びキャッシュ用ディスク手段と、ネットワーク上のＷＷＷデータの転送のログ・データを集計・解析する集計・解析手段と、ユーザからの指示を入力する入力手段と、該入力手段及び該集計・解析手段の内容を基に該ＷＷＷデータを該キャッシュ用メモリ手段又は該キャッシュ用ディスク手段により記憶するか否かを判定し、該ＷＷＷデータの該キャッシュ用メモリ手段又は該キャッシュ用ディスク手段による書き込み及び読み出し、又は上位ＷＷＷプロキシ装置又はＷＷＷサーバ装置から該ＷＷＷデータの取得を行うキャッシュ判定手段とを具備するものである。

【００１０】さらに、該キャッシュ判定手段が、通常は記憶すべき該ＷＷＷデータを該キャッシュ用メモリ手段にキャッシュし、該キャッシュ用メモリ手段のキャッシュデータの置換により溢れたＷＷＷデータについて、一定条件を満たすＷＷＷデータのみ該キャッシュ用ディスク手段にキャッシュすることを特徴とするものである。

【００１１】さらに、本発明は、該ＷＷＷプロキシ装置が、該集計・解析手段により集計されたログ・データのうち、最近の一定期間のログ・データのみのアクセス頻度の上位のＷＷＷデータのみキャッシュ処理を行うことを特徴とするものである。

【００１２】さらに、本発明は、該キャッシュ用メモリ手段のキャッシュデータの置換により溢れたＷＷＷデータを該キャッシュ用ディスク手段にキャッシュするか否かを判定する上記一定条件が、ネットワーク上のＷＷＷデータの転送のログ・データを集計・解析する集計・解析手段により集計された該ＷＷＷデータのアクセス頻度の上位であることを特徴とするものである。

【００１３】さらに、本発明は、該キャッシュ用メモリ手段のキャッシュデータの置換により溢れたＷＷＷデータを該キャッシュ用ディスク手段にキャッシュするか否かを判定する上記一定条件が、ユーザからの指示を入力する該入力手段の入力内容であることを特徴とするものである。

【００１４】さらに、本発明は、該ＷＷＷプロキシ装置が、ＷＷＷプロキシ装置を制御管理するためのデータベースを具備し、該データベースを用いて該ＷＷＷプロキシ装置を制御管理する制御管理手段を該ＷＷＷプロキシ装置内部又は外部に具備し、該制御管理手段が、必要に応じて該キャッシュ判定手段の処理を中止する指示を行うことにより該キャッシュ判定手段を制御管理することを特徴とするものである。

【００１５】さらに、本発明は、該ＷＷＷプロキシ装置が、ＷＷＷブラウザから代理取得を依頼されたＷＷＷデータを上位ＷＷＷプロキシ装置にさらに分散して代理処理させる分散制御手段を具備し、該分散制御手段が代理処理させる上位ＷＷＷプロキシ装置へのリクエストを該キャッシュ判定手段に指示することにより該キャッシュ判定手段を制御することを特徴とするものである。

【００１６】さらに、本発明は、該ＷＷＷプロキシ装置及び該上位ＷＷＷプロキシ装置が、ＷＷＷプロキシ装置を制御管理するためのデータベースを具備し、該データベースを用いて該ＷＷＷプロキシ装置及び該上位ＷＷＷプロキシ装置を制御管理する制御管理手段を該ＷＷＷプロキシ装置内部又は外部に具備し、該制御管理手段が、必要に応じて該ＷＷＷプロキシ装置及び該上位ＷＷＷプロキシ装置の該キャッシュ判定手段の処理を中止する指示、又は該ＷＷＷプロキシ装置及び該上位ＷＷＷプロキシ装置の該データベースの動的な変更を行うことにより該ＷＷＷプロキシ装置及び該上位ＷＷＷプロキシ装置の該キャッシュ判定手段を制御管理することを特徴とするものである。

【００１７】さらに、本発明は、ＷＷＷブラウザから代理取得を依頼されたＷＷＷデータを上位ＷＷＷプロキシ装置にさらに分散して代理処理させる該分散制御手段が、該ＷＷＷデータのURLのドメインネーム部分をハッシュ関数を用いて数値に変換する数値変換手段と、該数値に対応する上位ＷＷＷプロキシ装置に該ＷＷＷデータの処理を依頼する分散処理指示手段とを具備し、該分散処理指示手段が、代理処理させる上位ＷＷＷプロキシ装置へのリクエストを該キャッシュ判定手段に指示することにより該キャッシュ判定手段を制御することを特徴とするものである。

【００１８】さらに、本発明は、該ＷＷＷプロキシ装置

又は該上位WWWプロキシ装置が、リクエストされたWWWデータの参照先のWWWデータの先読みを行う先読み手段を具備し、該先読み手段が、該キャッシュ判定手段にリクエストの指示を行うことにより該キャッシュ判定手段を制御することを特徴とするものである。

【００１９】さらに、本発明は、リクエストされたWWWデータの参照先のWWWデータの先読みを行う該先読み手段が、一定条件を満たす該WWWデータのみ先読みを行う先読み制御手段を具備し、該先読み制御手段が、該キャッシュ判定手段にリクエストの指示を行うことにより該キャッシュ判定手段の先読み制御を行うことを特徴とするものである。

【００２０】さらに、本発明は、リクエストされたWWWデータの参照先のWWWデータの先読みの制御を行う該先読み制御手段が、ネットワーク上のWWWデータの転送のログ・データを集計・解析する集計・解析手段により集計されたアクセス頻度が上位である該WWWデータのみリクエストする指示を該キャッシュ判定手段に行うことにより該キャッシュ判定手段の先読みの制御を行うことを特徴とするものである。

【００２１】さらに、本発明は、リクエストされたWWWデータの参照先のWWWデータの先読み行う該先読み手段が、WWWデータをリクエストしたユーザの重要度を判定する重要度判定手段を具備し、該重要度判定手段は、各ユーザの重要度を記述した重要度データベースを具備し該重要度データベースを用いて該ユーザの重要度を判定することを特徴とし、該先読み制御手段が、該重要度判定手段による判定結果を用いて該キャッシュ判定手段にリクエストの指示を行うことにより該キャッシュ判定手段の先読みの制御を行うことを特徴とするものである。

【００２２】さらに、本発明は、該先読み制御手段が、該重要度判定手段による判定結果、及び該集計・解析手段により集計されたアクセス頻度を用いて該キャッシュ判定手段にリクエストの指示を行うことにより該キャッシュ判定手段の先読みの制御を行うことを特徴とするものである。

【００２３】通常は記憶すべきWWWデータを、キャッシュ判定手段によりキャッシュ用メモリ手段にキャッシュし、キャッシュ用メモリ手段のキャッシュデータの置換により溢れたWWWデータについて、集計・解析手段の内容、又は入力手段による入力内容に該当するWWWデータのみキャッシュ用ディスク手段にキャッシュすることにより、高速でキャッシュヒット率の高いキャッシュ処理を行うことができるという作用がある。

【００２４】また、リクエストされたWWWデータの参照先のWWWデータの先読みを行う先読み手段が、集計・解析手段の内容に該当するWWWデータのみ先読みの指示を行うことにより、キャッシュヒット率の高いキャッシュ処理とネットワークの負荷を低減できるという作用がある。

【００２５】

【発明の実施の形態】以下、本発明の実施の形態の一例を図面に基づき具体的に説明する。

【００２６】まず、本発明のWWWプロキシ装置の物理構成例を図１５を用いて説明する。

【００２７】図１５の物理構成図において、本発明のWWWプロキシ装置１５０１は、全ての処理プログラムを実行するプロセッサ１５０２、処理プログラムの実行に用いるデータを一時的に記憶しておくメモリ装置１５０３、プログラムやデータ等を格納しておくディスク装置１５０４、ユーザからの入力を装置内部に取り込む入力装置１５０５、外部とのWWWデータ及び制御信号の送受信を行う通信制御装置１５０６、キャッシュすべきWWWデータを記憶しておくキャッシュ用メモリ装置１５０７、キャッシュすべきWWWデータを格納しておくキャッシュ用ディスク装置１５０８、WWWプロキシ装置の動作状況、負荷状況、及び上位WWWプロキシ装置の数とアドレスを記録している管理制御のためのデータベースを用いて、WWWプロキシ装置及び上位WWWプロキシ装置を常に監視する制御管理装置１５０９を備え、制御管理装置１５０９以外の装置は、内部通信バス１５１０を介してWWWプロキシ装置１５０１内部で相互に接続される。

【００２８】また、通信制御装置１５０６は、外部通信バス１５１１を介してインターネット等の外部ネットワークに接続される。制御管理装置１５０９は、外部通信バス１５１１を介して通信制御装置１５０６と接続される。

【００２９】次に、本発明におけるWWWプロキシ装置の原理について、図１，図２、図３、図４、及び図１５を用いて説明する。

【００３０】図１は本発明によるWWWプロキシ装置の原理を示す図であり、図２は本発明によるWWWプロキシ装置と本発明によるWWWプロキシ装置を複数用いて行う分散処理の簡単な原理を示す図である。

【００３１】WWWブラウザ１０１は、WWW上のハイパーテキストや画像データなどのWWWデータを取得し表示するWWWクライアントであり、外部通信バス１５１１を介して通信制御装置１５０６と接続される。

【００３２】WWWプロキシ装置１０２は、一般にWWWブラウザ１０１の代理でインターネット等１００へリクエストを行う装置である。

【００３３】キャッシュ判定手段１０３は、WWWブラウザ１０１からのリクエストを受けて、リクエストされたWWWデータがキャッシュ用メモリ手段１０４又はキャッシュ用ディスク手段１０５によってキャッシュされているか否かを判定し、キャッシュされている場合はそのWWWデータをWWWブラウザ１０１に返し、キャッシュされていない場合はインターネット等１００にある

ＷＷＷサーバからＷＷＷデータを取得し、ＷＷＷブラウザ１０１にＷＷＷデータを返すと同時に取得したＷＷＷデータをキャッシュ用メモリ手段によって、又はキャッシュ用メモリ手段のキャッシュデータの置換により溢れたＷＷＷデータをキャッシュ用ディスク手段によってキャッシュするか否かを判定し、キャッシュする場合はキャッシュ用メモリ手段１０４又はキャッシュ用ディスク手段１０５にキャッシュする指示を行う手段であり、ディスク装置１５０４内に処理プログラムを持ち、プロセッサ１５０２が実際の処理を行う。

【００３４】キャッシュ用メモリ手段１０４は、キャッシュ判定手段１０３の指示により、キャッシュ用メモリ内にＷＷＷデータを格納したり、キャッシュ用メモリ内に格納しているＷＷＷデータを渡したりする手段であり、ディスク装置１５０４内に処理プログラムを持ち、プロセッサ１５０２が実際の処理を行い、通信制御装置１５０６を介して受信したＷＷＷデータをメモリ装置１５０３へ転送し、LRUによるキャッシュデータ置換により溢れたＷＷＷデータを一旦メモリ装置１５０３へ転送した後、メモリ装置１５０３内の受信したＷＷＷデータをキャッシュ用メモリ装置１５０７内に記憶し、また、キャッシュ用メモリ装置１５０７内のＷＷＷデータを通信制御装置１５０６に転送する。

【００３５】キャッシュ用ディスク手段１０５は、キャッシュ判定手段１０３の指示により、キャッシュ用メモリ手段１０４のキャッシュデータ置換により溢れたＷＷＷデータをキャッシュ用ディスク内に格納したり、キャッシュ用ディスク内に格納しているＷＷＷデータをキャッシュ判定手段１０３に渡したりする手段であり、ディスク装置１５０４内に処理プログラムを持ち、プロセッサ１５０２が実際の処理を行い、通信制御装置１５０６を介して受信したＷＷＷデータをメモリ装置１５０３へ転送し、最終的にキャッシュ用ディスク装置１５０８内に記憶し、また、キャッシュ用ディスク装置１５０８内のＷＷＷデータを通信制御装置１５０６に転送する。

【００３６】集計・解析手段１０６は、ＷＷＷブラウザ１０１からリクエストされるＷＷＷデータのログ・データをキャッシュ判定手段１０３から受け、ログ・データを集計・解析しアクセス頻度として保持しておく手段であり、ディスク装置１５０４内に処理プログラム及びアクセス頻度データを持ち、プロセッサ１５０２が実際の処理を行う。

【００３７】さらに、集計・解析手段１０６は、最近の一定期間のみのログ・データについても集計し、最近のアクセス頻度として保持しておき、通常は全てのログ・データにより集計されたアクセス頻度を用い、キャッシュ処理の条件として最近のアクセス頻度を用いるように予め設定されている場合には、最近のアクセス頻度も使用する。

【００３８】入力手段１０７は、キャッシュすべきＷＷＷデータのURL(Uniform Resource Locator)やドメイン・ネーム等をユーザからの指示により保持し、その内容をキャッシュ判定手段１０３に指示する手段であり、入力装置１５０５により処理を行い、入力されたデータをディスク装置１５０４内に保持しておく。

【００３９】分散制御手段１０８は、ＷＷＷブラウザ１０１からのリクエストをさらに複数の上位ＷＷＷプロキシ装置２００に分散して処理を依頼するようにキャッシュ判定手段１０３に対して指示を行う手段であり、ディスク装置１５０４内に処理プログラムを持ち、プロセッサ１５０２が実際の処理を行う。

【００４０】データベース１０９は、ＷＷＷプロキシ装置１０１の負荷や動作状況、上位ＷＷＷプロキシ装置２００の数及びアドレスを記録した管理情報ベースであり、ディスク装置１５０４内に保持され、通信制御装置１５０６を介して制御管理装置１５０９が読み出し及び書き込みを行う。

【００４１】制御管理手段１１０は、データベース１０９を用いてＷＷＷプロキシ装置１０１の動作状況を常に監視し、負荷が高い、又は障害が発生しているＷＷＷプロキシ装置１０２のキャッシュ判定手段１０３に対し処理中止、又は処理依頼中止、及び上位ＷＷＷプロキシ装置２００の変更の指示を行う手段であり、制御管理装置１５０９内にある。

【００４２】先読み手段１１１は、キャッシュ判定手段１０３がインターネット等１００から取得してきたＷＷＷデータを受け取り、そのＷＷＷデータがハイパーテキストである場合に参照しているURLを読みとり、そのURLで示されるＷＷＷデータを取得する指示をキャッシュ判定手段１０３に対して行う手段であり、ディスク装置１５０４内に処理プログラムを持ち、取得したＷＷＷデータをメモリ装置１５０３に一時記憶し、プロセッサ１５０２が、ＷＷＷデータからのURL読みとり、ＷＷＷデータ取得指示の処理を行う。

【００４３】記憶すべきＷＷＷデータを通常はキャッシュ用メモリ手段にキャッシュし、キャッシュ用メモリ手段のキャッシュデータの置換により溢れたＷＷＷデータについて、一定条件を満たすＷＷＷデータのみキャッシュ用ディスク手段にキャッシュすることにより、高速なキャッシュ処理の実現と、キャッシュのヒット率を向上させることが可能となる。

【００４４】また、最近の一定期間のログ・データのアクセス頻度の上位のＷＷＷデータのみキャッシュ処理を行うことにより、ユーザの取得したいＷＷＷデータの傾向が変化した場合でも、その変化をキャッシュしているＷＷＷデータに反映させることができ、結果としてキャッシュのヒット率の向上と、ネットワークの負荷低減が可能となる。

【００４５】また、キャッシュ用メモリ手段のキャッシュデータの置換により溢れたＷＷＷデータについて、ア

クセス頻度の上位のWWWデータのみキャッシュ用ディスク手段にキャッシュすることにより、リクエストされる可能性の高いWWWデータを保持でき、結果としてキャッシュのヒット率の向上と、ネットワークの負荷低減が可能となる。

【００４６】また、キャッシュ用メモリ手段のキャッシュデータの置換により溢れたWWWデータについて、ユーザからの指示を入力する入力手段の入力内容に合致するWWWデータのみキャッシュ用ディスク手段にキャッシュすることにより、ユーザが取得したいWWWデータを保持しておくことができ、結果としてキャッシュのヒット率の向上と、ネットワークの負荷低減が可能となる。

【００４７】また、制御管理手段がWWWプロキシ装置の動作状況、負荷状況を常に監視し、WWWプロキシ装置の負荷が高い場合に適宜処理を中止させる指示を行うことにより、WWWプロキシ装置の障害を未然に防ぐことが可能となる。

【００４８】また、リクエストされたWWWデータの参照先のWWWデータの先読みを行うことにより、ユーザが引き続いて参照先のリクエストを行った場合に、該当するWWWデータを高速に返信することが可能となる。

【００４９】また、分散制御手段によって複数の上位WWWプロキシ装置にさらに分散して処理を依頼することにより、WWWプロキシ装置の負荷分散と、キャッシュのヒット率の向上及びネットワークの負荷低減が可能となる。

【００５０】また、制御管理手段がWWWプロキシ装置及び上位WWWプロキシ装置の動作状況、負荷状況を常に監視し、WWWプロキシ装置及び上位WWWプロキシ装置の負荷が高い場合に適宜処理を中止するだけでなく、制御管理のためのデータベースを動的に変更し、負荷の高い上位WWWプロキシ装置には予め処理の依頼を行わないよう指示することにより、より効率的な分散処理が可能となる。

【００５１】図３は本発明によるWWWプロキシ装置の分散制御手段に関する原理を示す図である。数値変換手段３０１は、WWWブラウザ１０１からのリクエストのURLをキャッシュ判定手段１０３から受け、そのURLのドメインネーム部をハッシュ関数により数値に変換し、その数値を分散処理指示手段３０２に渡す手段であり、ディスク装置１５０４内に処理プログラムを持ち、プロセッサ１５０２が実際の処理を行う。

【００５２】分散処理指示手段３０２は、数値変換手段３０１から受けた数値に対応するWWWプロキシ装置を選択し、選択されたWWWプロキシ装置に代理処理をするようにキャッシュ判定手段１０３に対して指示を行う手段であり、ディスク装置１５０４内に処理プログラムを持ち、プロセッサ１５０２が実際の処理を行う。

【００５３】分散処理指示手段によって分散処理の制御・管理をすることにより、分散処理依頼の高速化と、URLによる上位WWWプロキシ装置の負荷分散が可能となる。

【００５４】図４は本発明による先読み手段に関する原理を示す図である。先読み制御手段４０１は、キャッシュ判定手段１０３の取得したWWWデータを読み込み、ハイパーテキストであるかどうかを判定し、ハイパーテキストである場合にその参照先のWWWデータのURLを解析し、さらに、集計・解析手段１０６又は重要度判定手段４０３からの指示により、先読みを行うようキャッシュ判定手段１０３に対して指示を行う手段であり、ディスク装置１５０４内に処理プログラムを持ち、プロセッサ１５０２が実際の処理を行う。

【００５５】重要度データベース４０２は、WWWブラウザ１０１によりリクエストを行うユーザの重要度を記述したデータベースであり、ディスク装置１５０４内に保持される。

【００５６】ここで、図１２は重要度データベース４０２の例を示す図であり、重要度データベース４０２は、図１２のようにユーザ情報１２０１に対して重要度１２０２の高低を記してある。ユーザ情報１２０１は、リクエストヘッダ内のユーザ情報や、プロキシ認証の際のユーザ情報等を用いる。次に、重要度判定手段４０３は、WWWブラウザ１０１によりリクエストを行うユーザの重要度を、重要度データベース４０２を用いて判定し、その判定結果を先読み制御手段４０１に渡す手段であり、ディスク装置１５０４内に処理プログラムを持ち、プロセッサ１５０２が実際の処理を行う。

【００５７】先読み制御手段が、リクエストしたユーザの重要度及びアクセス頻度を考慮し、重要であると判定されたユーザのリクエストしたWWWデータの参照先のWWWデータの中で、アクセス頻度の上位のWWWデータのみ先読みすることにより、ユーザの重要度に応じたサービスの提供とリクエストされる可能性の高いWWWデータのみの先読みによるネットワークの負荷低減が可能となる。

【００５８】以下、本発明によるWWWプロキシ装置の動作を、図１のようにWWWプロキシ装置を単独で用いる場合と、図２のように複数用いる場合について説明する。

【００５９】以降、全ての場合において、制御管理手段１１０は、WWWプロキシ装置１０２の負荷・動作状況を常に監視し、負荷が上昇していると判断した場合は該当する手段に対して処理の中止を指示し、又は該当する手段を有する上位WWWプロキシ装置２００に対する処理依頼を行わないようデータベース１０９を変更し、必要に応じてWWWブラウザ１０１にエラーメッセージを返すものとする。

【００６０】図１のようにWWWプロキシ装置を単独で用いる場合、処理の流れは図５のようになる。

【００６１】まず、ＷＷＷブラウザ１０１がＷＷＷプロキシ装置１０２に対してリクエストを行うと、キャッシュ判定手段１０３がリクエストを受信する（ステップＳ５０１）。キャッシュ判定手段１０３は、リクエストされたＷＷＷデータのＵＲＬを集計・解析手段１０６に渡し、アクセス頻度として集計させる（ステップＳ５０２）。次に、キャッシュ判定手段１０３は、リクエストされたＷＷＷデータがキャッシュ用メモリ手段１０４及びキャッシュ用ディスク手段１０５によってキャッシュされているかキャッシュ用メモリ手段１０４及びキャッシュ用ディスク手段１０５に問い合わせる（ステップＳ５０３）。

【００６２】ステップＳ５０３において、キャッシュされていると判定された場合、キャッシュ判定手段１０３は、該当するＷＷＷデータをＷＷＷブラウザ１０１に送信する（ステップＳ５０６）。さらに、リクエストされたＷＷＷデータについて先読み処理を行い（ステップＳ５０７）、終了する。ステップＳ５０７の先読み処理については、図７により後述する。

【００６３】ステップＳ５０３において、キャッシュされていないと判定された場合、キャッシュ判定手段１０３は、リクエストされたＷＷＷデータをインターネット等１０１から取得する（ステップＳ５０４）。次に、キャッシュ判定手段１０３は、取得したＷＷＷデータのキャッシュ処理を行う（ステップＳ５０５）。ステップＳ５０５のキャッシュ処理については、図６により後述する。

【００６４】さらに、キャッシュ判定手段１０３は、取得したＷＷＷデータをＷＷＷブラウザ１０１に送信し（ステップＳ５０６）、取得したＷＷＷデータについて先読み処理を行い（ステップＳ５０７）、終了する。

【００６５】図５におけるキャッシュ処理手順（ステップＳ５０５）について、図６により説明する。

【００６６】まず、キャッシュ判定手段１０３は、集計・解析手段１０６に最近の一定期間のアクセス頻度をキャッシュ処理の条件として用いるか否かを問い合わせ（ステップＳ６０１）、最近の一定期間のアクセス頻度をキャッシュ処理の条件として用いる場合、取得したＷＷＷデータのＵＲＬが最近の一定期間のアクセス頻度の上位のＵＲＬであるかを集計・解析手段１０６に問い合わせる（ステップＳ６０２）。もし、取得したＷＷＷデータのＵＲＬが最近のアクセス頻度の上位のＵＲＬでなければＷＷＷデータのキャッシュ処理は行わず、終了する。

【００６７】ここで、最近の一定期間を７日間とした場合のアクセス頻度の集計方法を図１１に示す。図１１の例では、日曜日から土曜日までの一日分のログ・データ１１０１～１１０７を一日毎に更新し（ステップＳ１１０８）、各曜日のログ・データ１１０１～１１０７を集計し（ステップＳ１１０９）、７日間のアクセス頻度１１１０とする。

【００６８】取得したＷＷＷデータのＵＲＬが最近の一定期間のアクセス頻度の上位であった場合、及びステップＳ６０１において、最近のアクセス頻度をキャッシュ処理の条件として用いない場合、キャッシュ用メモリ手段１０４はＬＲＵキャッシュデータ置換アルゴリズムに基づき最後に参照されてから最も古いＷＷＷデータをキャッシュ判定手段１０３に転送すると同時にキャッシュ用メモリ内から削除し、キャッシュ判定手段１０３は、取得したＷＷＷデータをキャッシュ用メモリ手段１０４に渡し、メモリ装置にキャッシュする（ステップＳ６０３）。

【００６９】次に、キャッシュ判定手段１０３は、キャッシュ用メモリ手段１０４のキャッシュデータの置換により溢れたＷＷＷデータをキャッシュ用ディスク手段にキャッシュする一定条件が、集計・解析手段１０６によるアクセス頻度であるか、又は入力手段１０７の入力内容であるかをそれぞれの手段に問い合わせる（ステップＳ６０４）。

【００７０】上記一定条件が、集計・解析手段１０６によるアクセス頻度によるものである場合、キャッシュ判定手段１０３は、キャッシュ用メモリ手段１０４のキャッシュデータの置換により溢れたＷＷＷデータのＵＲＬがアクセス頻度の上位のＵＲＬであるかを集計・解析手段１０６に問い合わせ（ステップＳ６０５）、アクセス頻度の上位のＵＲＬであればキャッシュ用メモリ手段１０４のキャッシュデータの置換により溢れたＷＷＷデータをキャッシュ用ディスク手段１０５に渡し、ディスク装置にキャッシュする。アクセス頻度の上位のＵＲＬでなければ終了する。

【００７１】また、上記一定条件が、入力手段１０７による入力内容であった場合、キャッシュ判定手段１０３は、キャッシュ用メモリ手段１０４のキャッシュデータの置換により溢れたＷＷＷデータのＵＲＬが入力内容に該当しているかを入力手段１０７に問い合わせ（ステップＳ６０６）、入力内容に該当していればキャッシュ用メモリ手段１０４のキャッシュデータの置換により溢れたＷＷＷデータをキャッシュ用ディスク手段１０５に渡し、ディスク装置にキャッシュする。入力内容に該当しなければ終了する。

【００７２】図５における先読み処理手順（ステップＳ５０７）について、図７により説明する。

【００７３】まず、先読み制御手段４０１は、キャッシュ判定手段１０３が取得したＷＷＷデータを読み込み、リクエストを行ったユーザの情報を重要度判定手段４０３に渡す。重要度判定手段４０３は、受け取ったユーザの情報を重要度データベース４０２で調べ（ステップＳ７０１）、ユーザの重要度を先読み制御手段に返す。先読み制御手段４０１は、ユーザの重要度が低い場合、先読みの処理は行わず終了し、ユーザの重要度が高い場合は、取得したＷＷＷデータがハイパーテキストであるか

を判定する（ステップＳ７０２）。
【００７４】取得したWWWデータがハイパーテキストでない場合、処理を終了し、ハイパーテキストである場合は、WWWデータが参照しているWWWデータのURLを順に読み込む（ステップＳ７０３）。次に、先読み制御手段４０１は、読み込んだURLが集計・解析手段１０６のアクセス頻度の上位のURLであるかを集計・解析手段１０６に問い合わせ（ステップＳ７０４）、アクセス頻度の上位でないと判定された場合は終了する。
【００７５】ステップＳ７０４において、上記URLがアクセス頻度の上位であると判定された場合、先読み制御手段４０１は、取得すべきWWWデータがキャッシュ用メモリ手段１０４、又はキャッシュ用ディスク手段１０５によりキャッシュされているかをキャッシュ判定手段１０３を介して問い合わせる（ステップＳ５０３）。
【００７６】ステップＳ５０３において、取得すべきWWWデータがキャッシュされていない場合、先読み制御手段４０１は、上記URLのWWWデータを取得する指示をキャッシュ判定手段１０３に行い（ステップＳ７０５）、キャッシュ判定手段は１０３は、インターネット等１００のサーバからWWWデータを取得し（ステップＳ７０６）、取得したWWWデータのキャッシュ処理を行う（ステップＳ５０５）。
【００７７】ステップＳ５０３において、取得すべきWWWデータがキャッシュされている場合、及びステップＳ５０５において、取得したWWWデータのキャッシュ処理が終了した場合、先読み制御手段４０１は、最初に取得したWWWデータの参照先のURLを順に調べ、先読みをするか否かの判定が行われていないWWWデータのURLがあるか調べる（ステップＳ７０７）。
【００７８】ステップＳ７０７において、全ての参照WWWデータについて先読みをするか否かの判定が行われていれば、先読み処理を終了する。先読みをするか否か未判定の参照WWWデータがあれば、ステップＳ７０３に戻り、それ以下の処理を繰り返す。
【００７９】図２又は図３のように、複数のWWWプロキシ装置１０２を階層的に用いる場合の処理の手順を図８に示す。
【００８０】WWWブラウザ１０１によりWWWプロキシ装置１０２に対してリクエストが行われると、キャッシュ判定手段１０３によりリクエストを受信する（ステップＳ５０１）。キャッシュ判定手段１０３は、リクエストされたWWWデータのURLを集計・解析手段１０６に渡し、アクセス頻度として集計させる（ステップＳ５０２）。次に、キャッシュ判定手段１０３は、リクエストされたWWWデータがキャッシュ用メモリ手段１０４及びキャッシュ用ディスク手段１０５によってキャッシュされているか問い合わせる（ステップＳ５０３）。
【００８１】ステップＳ５０３において、キャッシュされていると判定された場合、該当するWWWデータをWWWブラウザ１０１に送信する（ステップＳ５０６）。さらに、リクエストされたWWWデータについて先読み処理を行い（ステップＳ５０７）、終了する。
【００８２】ステップＳ５０３において、キャッシュされていないと判定された場合、キャッシュ判定手段１０３は、リクエストされたWWWデータの取得を上位WWWプロキシ装置２００の一つに依頼する（ステップＳ８０１）。ステップＳ８０１については後述する。
【００８３】次に、キャッシュ判定手段１０３は、WWWデータの取得を依頼した上位WWWプロキシ装置３０３のキャッシュ判定手段１０３からWWWデータを取得し（ステップＳ８０２）、取得したWWWデータのキャッシュ処理を行う（ステップＳ５０５）。以降は、図５と同様に、取得したWWWデータをWWWブラウザ１０１に送信し（ステップＳ５０６）、先読み処理を行い（ステップＳ５０７）、終了する。
【００８４】次に、図８におけるステップＳ８０１について、図９により詳述する。まず、キャッシュ判定手段１０３は、リクエストされたWWWデータのURLを分散制御手段１０８内の数値変換手段３０１に渡す（ステップＳ９０１）。数値変換手段３０１は、受け取ったURLのドメインネーム部にハッシュ関数を適用し、数値に変換する（ステップＳ９０２）。
【００８５】ここで、数値に変換する方法は幾つか考えられるが、一例としてここではURLの文字コードの値を単純に加算するものとする。数値変換手段３０１は、変換した数値を分散処理指示手段３０２に渡す（ステップＳ９０３）。分散処理指示手段３０２は、受け取った数値に対応する上位WWWプロキシ装置２００の一つを選択する（ステップＳ９０４）。ここで、上位WWWプロキシ装置２００の選択の方法は幾つか考えられるが、一例としてここでは受け取った数値を上位WWWプロキシ装置２００の個数で割り、余り値により上位WWWプロキシ装置を対応させるものとする。
【００８６】例えば、図１３のように上位のWWWプロキシ装置１３０１～１３０４にそれぞれ０、１、２、３と数字を割り当てておき、数値変換手段３０１が、キャッシュ判定手段１０３からURLを受け（ステップＳ１３０５）、受け取ったURLを数値の１３に変換した（ステップＳ１３０６）とした場合、分散処理指示手段３０２に数値の１３を渡し（ステップＳ１３０７）、分散処理指示手段３０２は余り値１を計算し（ステップＳ１３０８）、１が割り当てられた上位WWWプロキシ装置に対してリクエストを行うように、キャッシュ判定手段１０３に指示する（ステップＳ１３０９）ことにより、上位WWWプロキシ装置の選択を行う。
【００８７】次に、分散処理指示手段３０２は、選択された上位WWWプロキシ装置３０３に対してWWWデータの取得のリクエストを行うようキャッシュ判定手段１０３に指示を行い、キャッシュ判定手段１０３は、選択

された上位WWWプロキシ装置３０３のキャッシュ判定手段１０３に対して、取得すべきWWWデータのリクエストを行う（ステップＳ９０５）。

【００８８】次に、選択された上位WWWプロキシ装置３０３は、リクエストされたWWWデータの取得を行う（ステップＳ９０６）。ステップＳ９０６については後述する。

【００８９】さらに、上位WWWプロキシ装置３０３は、取得したWWWデータを下位のWWWプロキシ装置１０２のキャッシュ判定手段１０３に送信し（ステップＳ９０７）、終了する。

【００９０】次に、図９におけるステップＳ９０６について、図１０により詳述する。WWWプロキシ装置１０２のキャッシュ判定手段１０３により行われたリクエストを、選択された上位のWWWプロキシ装置３０３は、WWWプロキシ装置３０３のキャッシュ判定手段１０３により受信する（ステップＳ１００１）。

【００９１】WWWプロキシ装置３０３のキャッシュ判定手段１０３は、リクエストされたWWWデータのURLをWWWプロキシ装置３０３の集計・解析手段１０６に渡し、アクセス頻度として集計させる（ステップＳ５０２）。次に、WWWプロキシ装置３０３のキャッシュ判定手段１０３は、リクエストされたWWWデータがWWWプロキシ装置３０３のキャッシュ用メモリ手段１０４及びキャッシュ用ディスク手段１０５によってキャッシュされているか問い合わせる（ステップＳ５０３）。

【００９２】ステップＳ５０３において、キャッシュされていると判定された場合、該当するWWWデータを下位のWWWプロキシ装置１０２のキャッシュ判定手段１０３に送信する（ステップＳ１００２）。さらに、リクエストされたWWWデータについて先読み処理を行い（ステップＳ５０７）、終了する。

【００９３】ステップＳ５０３において、キャッシュされていないと判定された場合、WWWプロキシ装置３０３のキャッシュ判定手段１０３は、リクエストされたWWWデータをインターネット等１００のWWWサーバから取得する（ステップＳ５０４）。次に、WWWプロキシ装置３０３のキャッシュ判定手段１０３は、取得したWWWデータのキャッシュ処理を行う（ステップＳ５０５）。

【００９４】さらに、WWWプロキシ装置３０３のキャッシュ判定手段１０３は、取得したWWWデータを下位のWWWプロキシ装置１０２のキャッシュ判定手段１０３に送信し（ステップＳ１００２）、取得したWWWデータについて先読み処理を行い（ステップＳ５０７）、終了する。

【００９５】

【発明の効果】本発明のWWWプロキシ装置によれば、キャッシュ装置としてメモリ装置及びディスク装置を併用することにより高速で効率的、且つ信頼性の高いキャッシュ処理を行うことができる。

【００９６】また、先読み処理については、ユーザがリクエストをする可能性の高いWWWデータのみ先読み処理することにより、より効率的なキャッシュ処理、及びネットワークの負荷の低減を実現することができる。

【図面の簡単な説明】

【図１】本発明によるWWWプロキシ装置の簡単な原理図。

【図２】本発明によるWWWプロキシ装置と分散処理の簡単な原理図。

【図３】本発明によるWWWプロキシ装置の分散制御手段に関する原理図。

【図４】本発明による先読み手段に関する原理図。

【図５】本発明によるWWWプロキシ装置を単独で用いる場合の処理手順の説明。

【図６】図５におけるキャッシュ処理手順（ステップＳ５０５）の説明。

【図７】図５における先読み処理手順（ステップＳ５０７）の説明。

【図８】本発明によるWWWプロキシ装置を複数用いた場合の処理手順の説明。

【図９】図８における上位WWWプロキシ装置への処理依頼手順（ステップＳ８０１）の説明。

【図１０】図９における選択された上位プロキシ装置によるWWWデータの取得処理手順（ステップＳ９０６）の説明。

【図１１】最近の一定期間のアクセス頻度の計算方法の例。

【図１２】重要度データベースの例。

【図１３】上位WWWプロキシ装置選択方法の一例を示す図。

【図１４】一般的なWWWプロキシ装置の役割を表す図。

【図１５】本発明におけるWWWプロキシ装置の物理構成の一例を示す図。

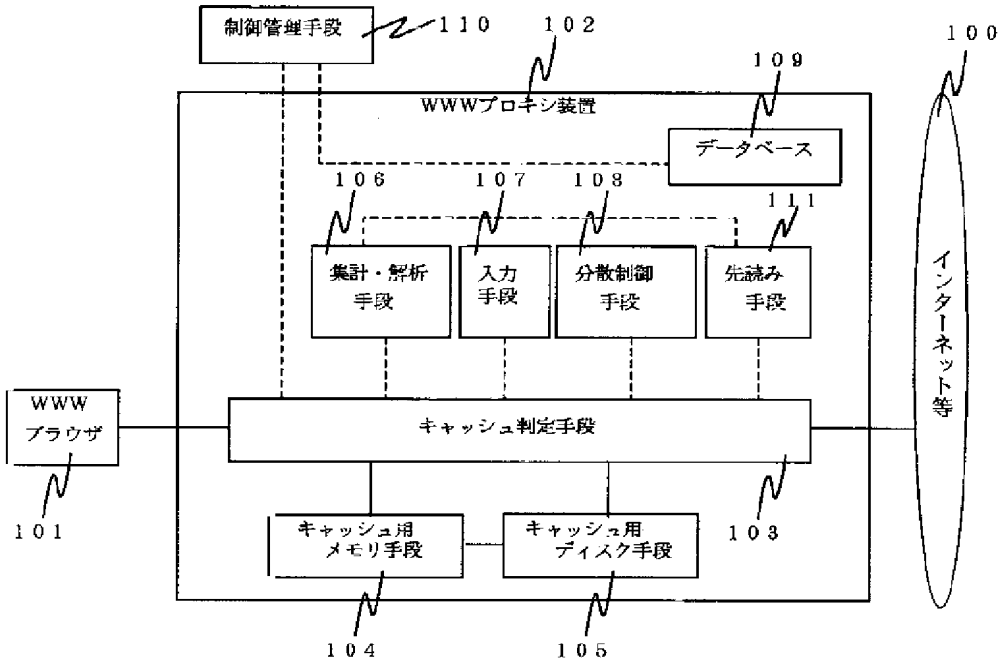
【符号の説明】

１００…インターネット等、１０１…WWWブラウザ、１０２…WWWプロキシ装置、１０３…キャッシュ判定手段、１０４…キャッシュ用メモリ手段、１０５…キャッシュ用ディスク手段、１０６…集計・解析手段、１０７…入力手段、１０８…分散制御手段、１０９…データベース、１１０…制御管理手段、１１１…先読み手段、２００…上位WWWプロキシ装置、３０１…数値変換手段、３０２…分散処理指示手段、３０３…選択された上位WWWプロキシ装置、４０１…先読み制御手段、４０２…重要度データベース、４０３…重要度判定手段、１３０１…上位WWWプロキシ装置（０を割り当て）、１３０２…上位WWWプロキシ装置（１を割り当て）、１３０３…上位WWWプロキシ装置（２を割り当て）、１３０４…上位WWWプロキシ装置（３を割り当て）、１

４０１…WWWクライアント、１４０２…WWWプロキシ、１４０３…キャッシュ装置、１４０４…インターネット、１４０５…WWWサーバ、１５０１…WWWプロキシ装置、１５０２…プロセッサ、１５０３…メモリ装置、１５０４…ディスク装置、１５０５…入力装置、１５０６…通信制御装置、１５０７…キャッシュ用メモリ装置、１５０８…キャッシュ用ディスク装置、１５０９…制御管理装置、１５１０…内部通信バス、１５１１…外部通信バス。

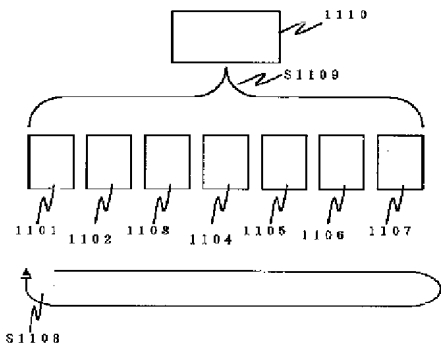
【図１】

図１



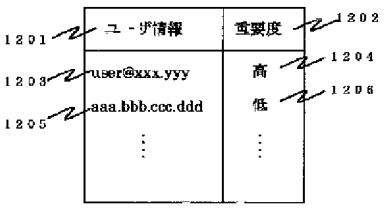
【図１１】

図１１



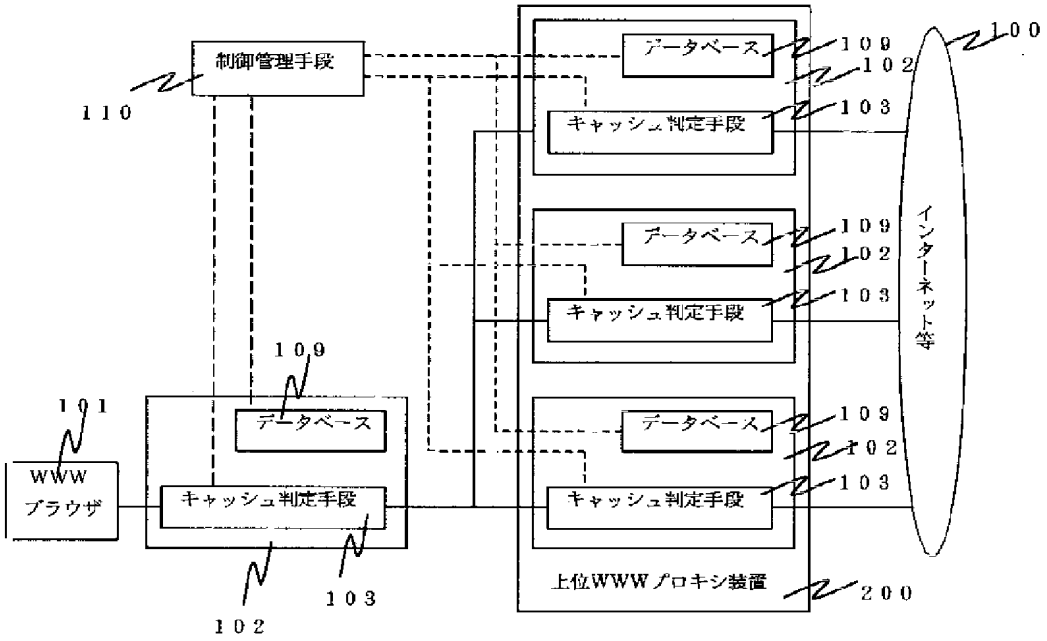
【図１２】

図１２



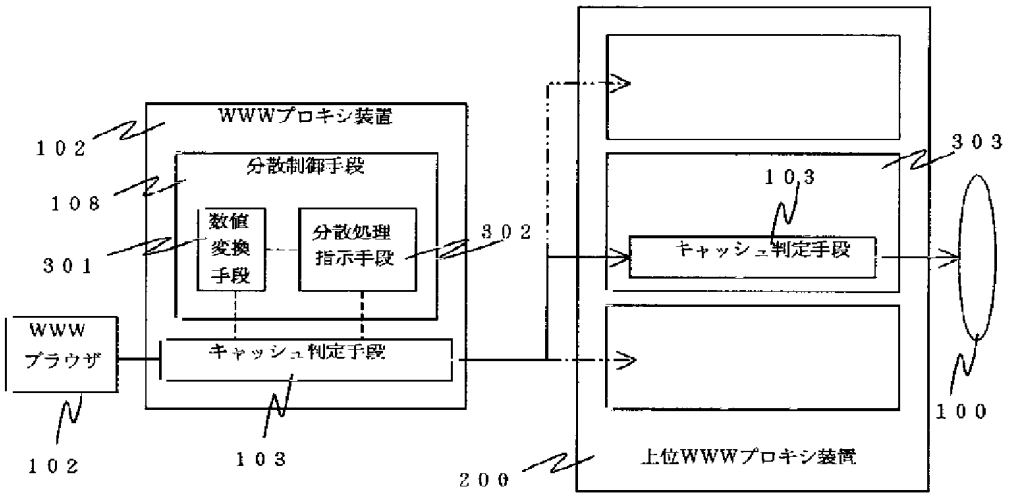
【図2】

図2


制御管理手段
110
データベース
109
102
キャッシュ判定手段
103
100
インターネット等
データベース
109
102
キャッシュ判定手段
103
109
101
データベース
WWW
ブラウザ
キャッシュ判定手段
102
103
データベース
109
102
キャッシュ判定手段
103
上位WWWプロキシ装置
200

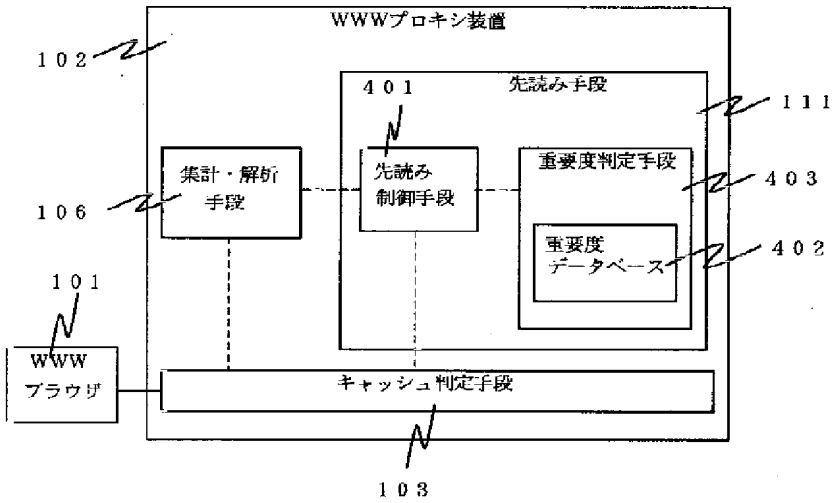
【図3】

図3


WWWプロキシ装置
102
分散制御手段
108
数値
変換
手段
301
分散処理
指示手段
302
WWW
ブラウザ
キャッシュ判定手段
102
103
303
103
キャッシュ判定手段
100
200
上位WWWプロキシ装置

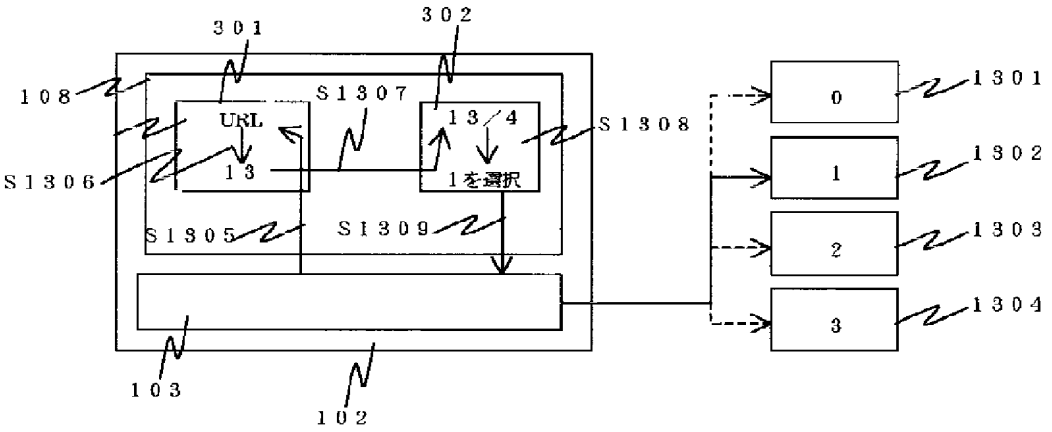
【図4】

図4


WWWプロキシ装置
102
401
先読み手段
111
集計・解析
手段
106
先読み
制御手段
重要度判定手段
403
重要度
データベース
402
101
WWW
ブラウザ
キャッシュ判定手段
103

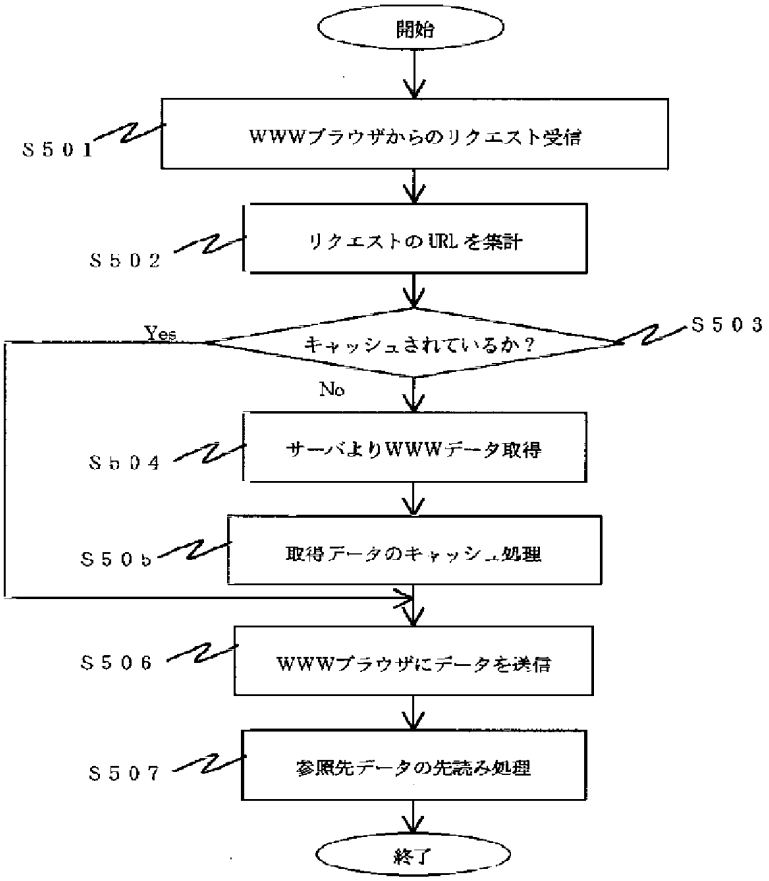
【図13】

図13


301
302
108
S1307
URL
13/4
S1308
S1306
13
1を選択
S1305
SI309
0
1301
1
1302
2
1303
3
1304
103
102

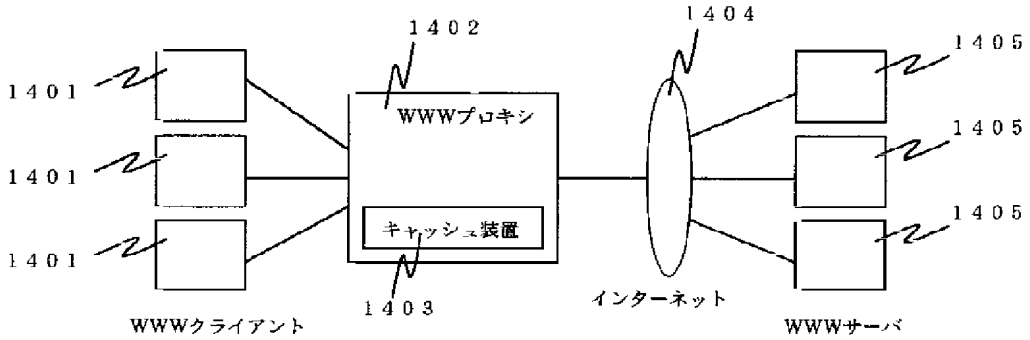
【図5】

図5


開始
S501
WWWブラウザからのリクエスト受信
S502
リクエストのURLを集計
Yes
キャッシュされているか？
S503
No
S504
サーバよりWWWデータ取得
S505
取得データのキャッシュ処理
S506
WWWブラウザにデータを送信
S507
参照先データの先読み処理
終了

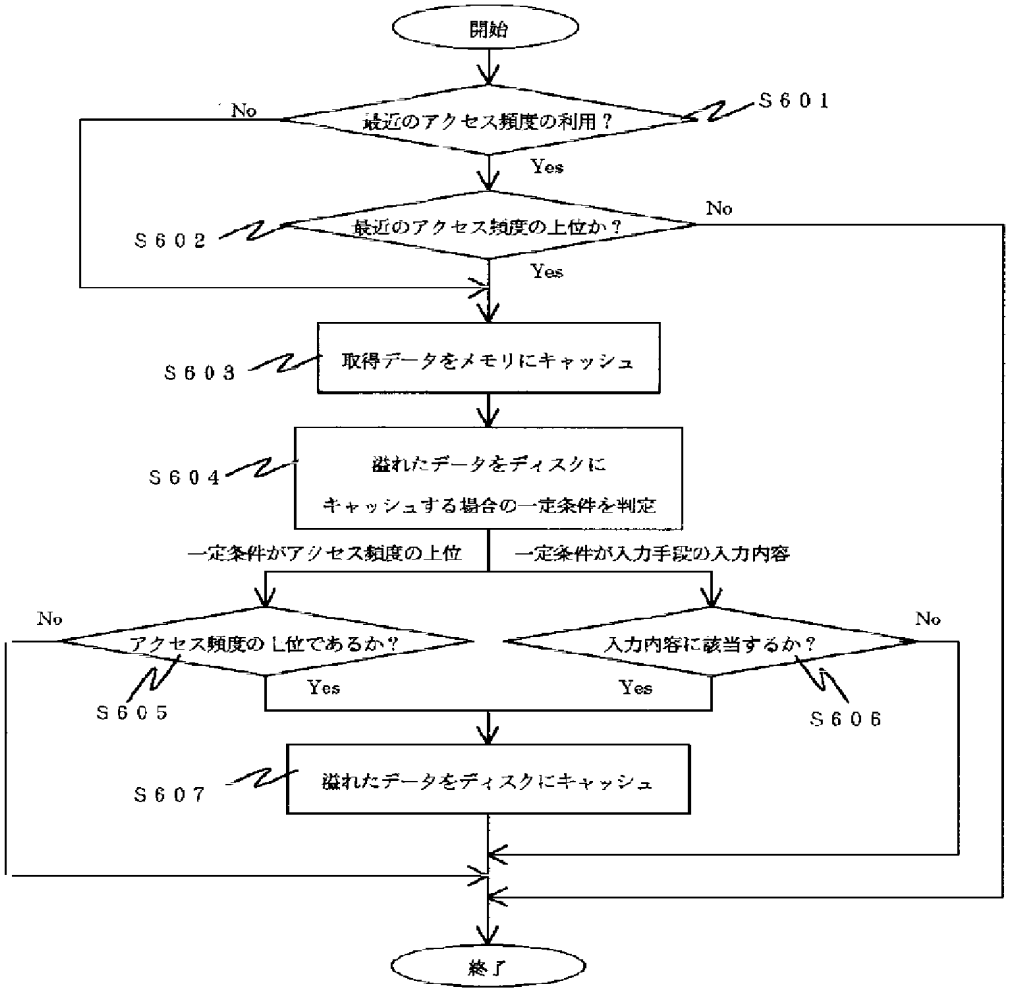
【図14】

図14


1402
1404
1405
1401
WWWプロキシ
1405
1401
キャッシュ装置
1405
1401
1403
インターネット
WWWクライアント
WWWサーバ

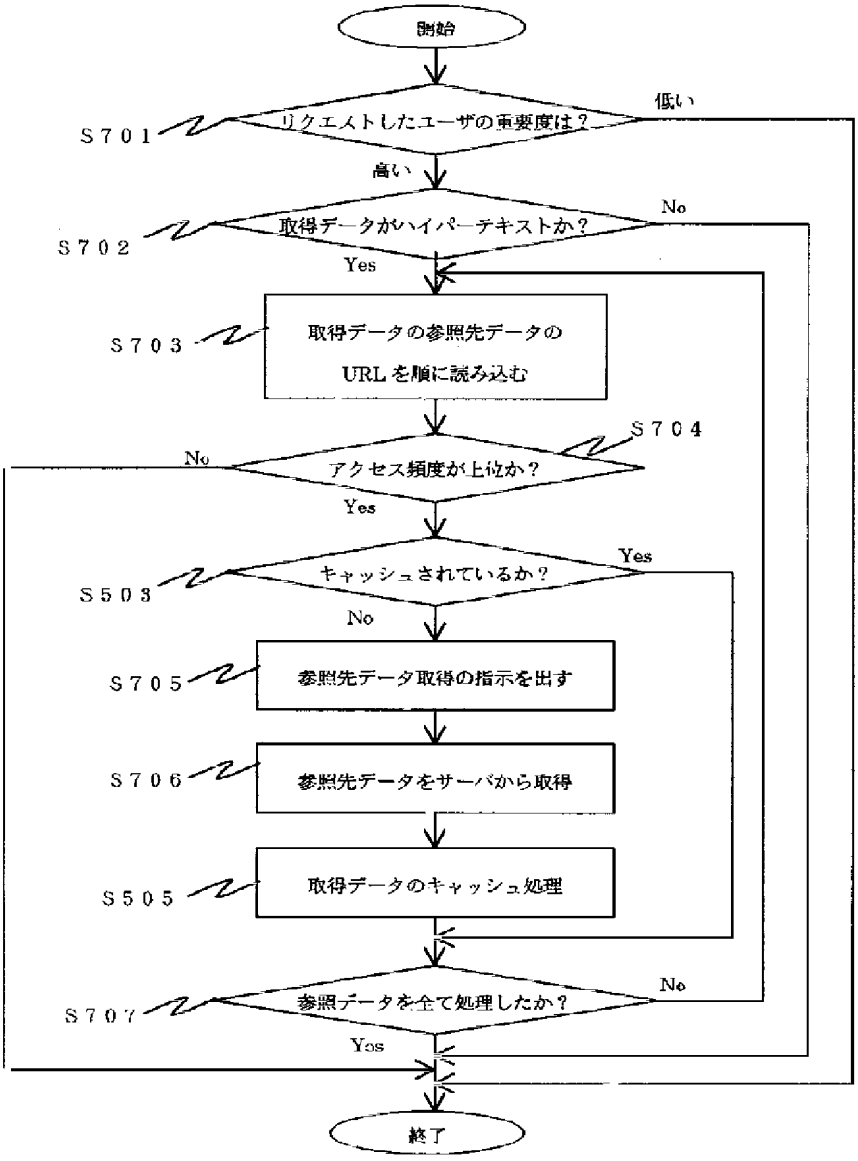
【図６】

図６


開始
No
最近のアクセス頻度の利用？
S601
Yes
最近のアクセス頻度の上位か？
No
S602
Yes
S603
取得データをメモリにキャッシュ
S604
溢れたデータをディスクに
キャッシュする場合の一定条件を判定
一定条件がアクセス頻度の上位
一定条件が入力手段の入力内容
No
アクセス頻度の上位であるか？
No
入力内容に該当するか？
Yes
Yes
S605
S606
S607
溢れたデータをディスクにキャッシュ
終了

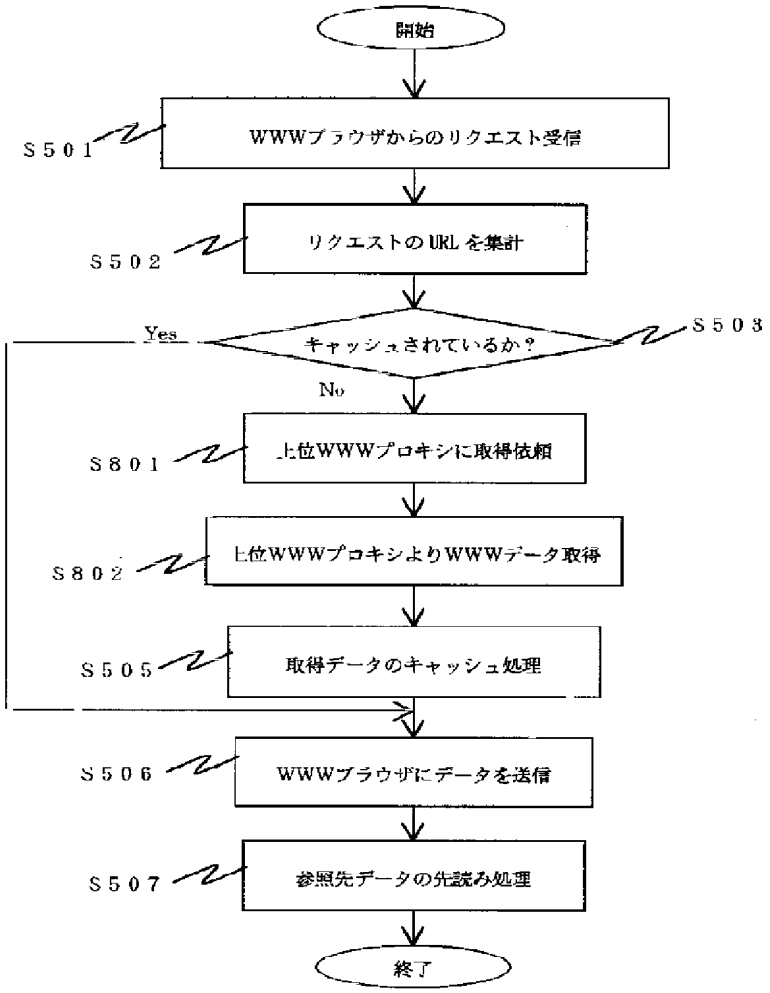
【図7】

図7


開始
S701
リクエストしたユーザの重要度は？
低い
高い
S702
取得データがハイパーテキストか？
No
Yes
S703
取得データの参照先データの
URLを順に読み込む
S704
No
アクセス頻度が上位か？
Yes
S503
キャッシュされているか？
Yes
No
S705
参照先データ取得の指示を出す
S706
参照先データをサーバから取得
S505
取得データのキャッシュ処理
S707
参照データを全て処理したか？
No
Yes
終了

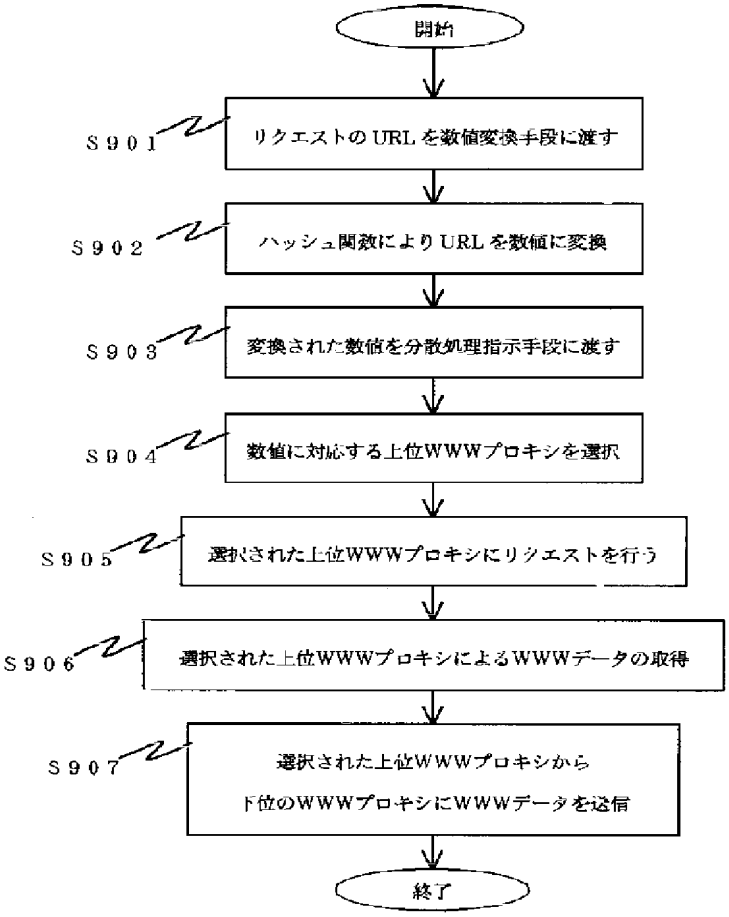
【図８】

図８


開始
S501
WWWブラウザからのリクエスト受信
S502
リクエストのURLを集計
Yes
キャッシュされているか？
S503
No
S801
上位WWWプロキシに取得依頼
S802
上位WWWプロキシよりWWWデータ取得
S505
取得データのキャッシュ処理
S506
WWWブラウザにデータを送信
S507
参照先データの先読み処理
終了

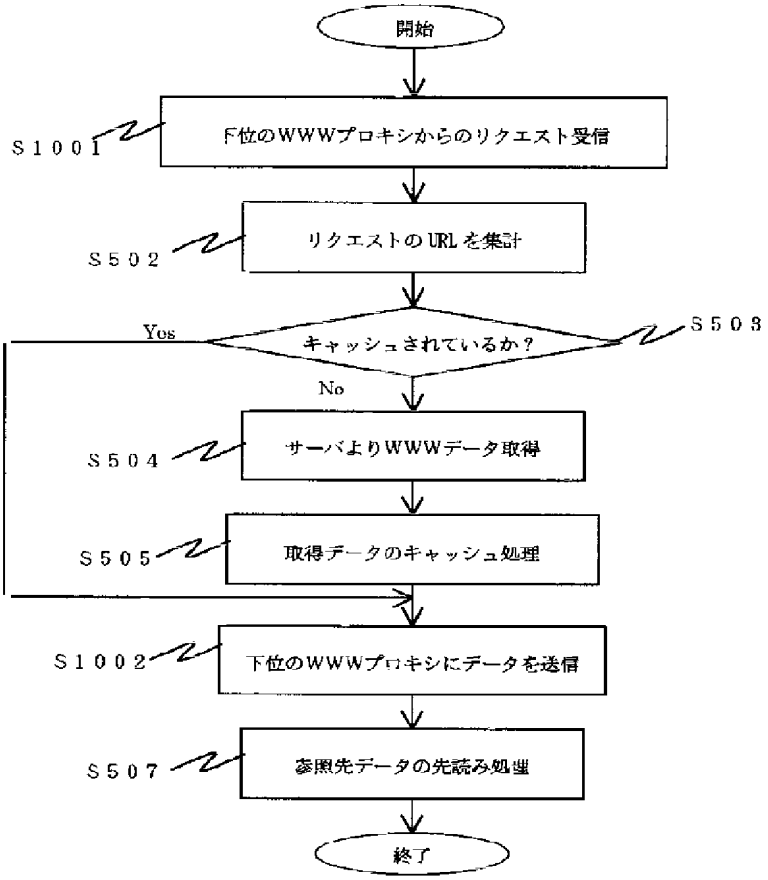
【図９】

図９


開始
Ｓ９０１
リクエストのURLを数値変換手段に渡す
Ｓ９０２
ハッシュ関数によりURLを数値に変換
Ｓ９０３
変換された数値を分散処理指示手段に渡す
Ｓ９０４
数値に対応する上位WWWプロキシを選択
Ｓ９０５
選択された上位WWWプロキシにリクエストを行う
Ｓ９０６
選択された上位WWWプロキシによるWWWデータの取得
Ｓ９０７
選択された上位WWWプロキシから
下位のWWWプロキシにWWWデータを送信
終了

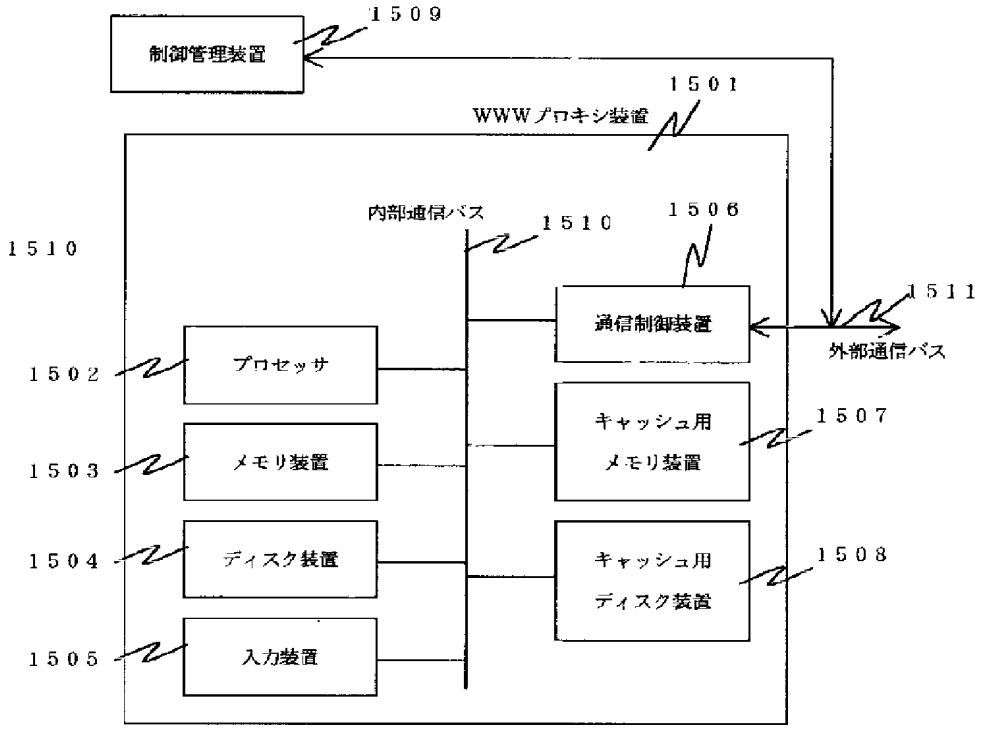
【図１０】

図１０


開始
S１００１
下位のWWWプロキシからのリクエスト受信
S５０２
リクエストのURLを集計
Yes
キャッシュされているか？
S５０３
No
S５０４
サーバよりWWWデータ取得
S５０５
取得データのキャッシュ処理
S１００２
下位のWWWプロキシにデータを送信
S５０７
参照先データの先読み処理
終了

【図１５】

図１５


１５０９
制御管理装置
１５０１
WWWプロキシ装置
内部通信バス
１５１０
１５０６
１５１０
１５１１
通信制御装置
外部通信バス
１５０２
プロセッサ
キャッシュ用
メモリ装置
１５０７
１５０３
メモリ装置
１５０４
ディスク装置
キャッシュ用
ディスク装置
１５０８
１５０５
入力装置

특2001-0023599

(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(51) Int. Cl.[7]
G06F 17/30

(11) 공개번호 특2001-0023599
(43) 공개일자 2001년03월26일

(21) 출원번호	10-2000-7002247		
(22) 출원일자	2000년03월02일		
번역문제출일자	2000년03월02일		
(86) 국제출원번호	PCT/GB1998/02920	(87) 국제공개번호	WO 1999/17227
(86) 국제출원출원일자	1998년09월28일	(87) 국제공개일자	1999년04월08일
(81) 지정국	EP 유럽특허 : 오스트리아 벨기에 스위스 독일 덴마크 스페인 프랑스 영국 그리스 아일랜드 이탈리아 룩셈부르크 모나코 네덜란드 포르투갈 스웨덴 핀랜드 사이프러스 국내특허 : 캐나다 중국 체코 헝가리 이스라엘 일본 대한민국 폴란드 러시아		
(30) 우선권주장	08/939,277 1997년09월29일 미국(US)		
(71) 출원인	인터내셔널 비지네스 머신즈 코포레이션 포만 제프리 엘 미국 10504 뉴욕주 아몬크		
(72) 발명자	말킨피터켄네스 미국10502뉴욕주아드슬리브램블브룩로드64 유필립시-룽 미국10514뉴욕주챔파큐스토너웨이18		
(74) 대리인	김창세, 김원준, 장성구		

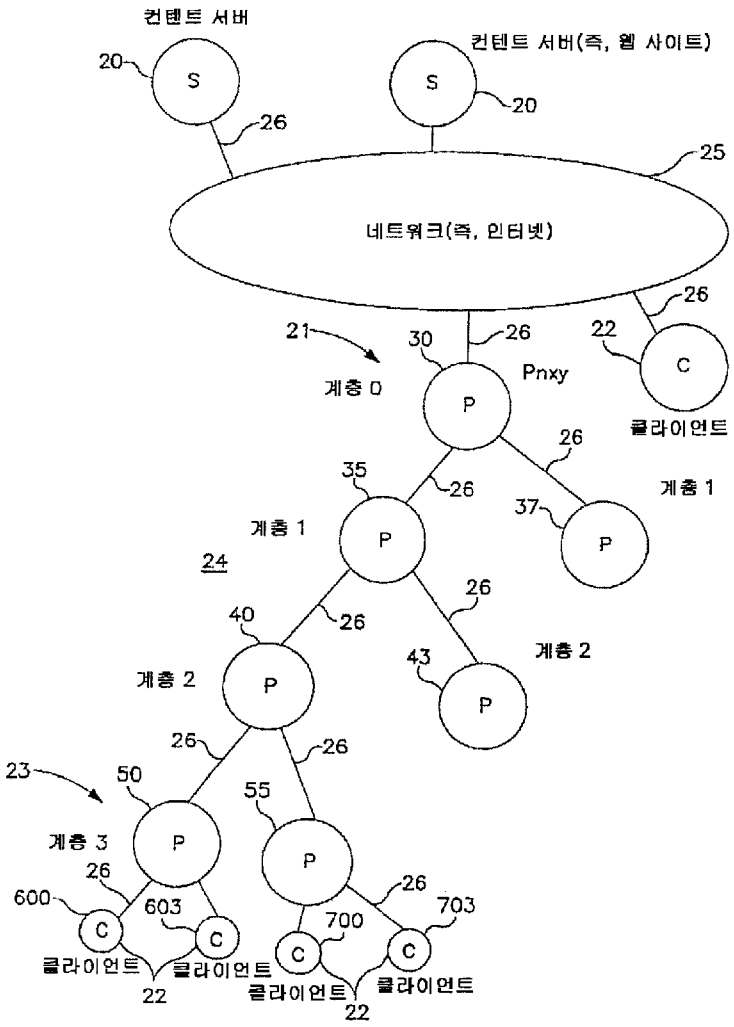
심사청구 : 있음

(54) 선택 정보 사전 인출 방법 및 장치

요약

본 발명은 프록시 서버 계층과 연관된 클라이언트를 위한 데이터 사전 인출 및 클라이언트로의 개선된 컨텐트 서버 데이터 전송(an enhanced content server data transfer)을 위한 방법 및 시스템에 관한 것이다. 데이터를 사전 인출하는 방법은 데이터 액세스 패턴을 식별하고 상기 데이터 액세스 패턴의 동적 해석에 기초하여 선택 정보를 사전 인출한다. 컨텐트 서버나 프록시 서버는 컨텐트 서버나 프록시 서버와 연관된 클라이언트의 데이터 액세스 조회 패턴을 식별한다. 데이터 액세스 패턴은 통계 데이블 내에 저장되는 것이 바람직하다. 컨텐트 서버는 상기 통계 테이블로부터의 입력에 기초하여 사전 인출 힌트 정보를 발생시킨다. 사전 인출 힌트 정보는 클라이언트에 의해 요청된 정보 또는 객체에 주석을 단다(annotate). 사전 인출 힌트 정보는 요청된 정보가 프록시 서버 계층을 통과할 때 갱신된다. 사전 인출 힌트 정보는 클라이언트에 의해 요청된 정보에 주석을 단다. 클라이언트에 대해 선택 정보를 사전 인출하는 궁극적인 판단은 사전 인출 힌트 정보와 사전 인출 힌트 값에 기초하여 행해진다. 사전 인출 힌트 값은 저장 제한 조건(storage constraints)이나 액세스 시간과 같은 지역 노드-고유 인자(local node-specific factors)에 의해 계산된다.

대표도



명세서

기술분야

본 발명은 클라이언트를 위한 정보 또는 객체의 동적 사전 인출(DYNAMIC PREFETCHING) 방법 및 장치에 관한 것으로서, 클라이언트는 월드 와이드 웹(World Wide Web) 상에서의 웹 객체의 사전 인출과 관련되는 것과 같이 프록시 서버의 계층과 연관되는 것이 바람직하다.

배경기술

용어 해설(GLOSSARY OF TERMS)

인터넷(Internet) : TCP/IP 짝(suite) 프로토콜 을 사용하는 네트워크나 게이트웨이(gateway)의 네트워크이다.

클라이언트(Client) : 커맨드와 관련된 작업을 수행하는 서버로 커맨드를 발행(issue)하는 컴퓨터이다.

서버(Server) : 다른 컴퓨터의 커맨드에 따라 작업을 수행하는 임의의 컴퓨터이다. 웹 서버는 전형적으로 하나 이상의 클라이언트를 지원한다.

월드 와이드 웹(World Wide Web : WWW 또는 Web) : 사용자가 (하이퍼링크(hyperlinks)로 알려진) 하이라

이트(highlight)된 관심 단어나 구(phrases)를 선택("클릭 온(click on)")함으로써, 서버에서 서버로, 대 이터베이스에서 데이터베이스로 인터넷 스위치 상에서 정보를 검색하도록 하는 인터넷 애플리케이션이다. 인터넷 WWW 서버는 클라이언트를 지원하고, 클라이언트에게 정보를 제공한다. 웹은 모든 자원이 어드레스(address)되는 인터넷으로 간주되며, HTML을 이용하여 URL에 대응하는 정보를 디스플레이하고, 다른 URL에 대해 포인트-및-클릭(point-and-click) 인터페이스를 제공한다.

URL(Universal Resource Locator) : 인터넷 상의 정보를 고유하게 식별하거나 어드레스하는 방법으로서, 전자 우편 어드레스의 웹 문서 버전(Web document version)으로 간주될 수 있다. URL은 하이퍼링크를 이용하여 액세스할 수 있다. URL 식별(identification)의 예로는 "http://www.hpilipyu.com:80/table.html"이 있다. URL은 4 가지 구성 요소를 갖는다. 이 예의 왼쪽부터 시작해서, 첫 번째 요소는 ":"에 의해 로케이터(locator)의 나머지 부분으로부터 구별되며 사용할 프로토콜(이 경우는 http)을 명시한다. 다음 요소는 목표 호스트의 호스트네임(hostname) 또는 IP 어드레스로서, 이 요소는 왼쪽은 이중 슬래쉬"//", 오른쪽은 단일 슬래쉬"/"(또는 선택적으로 ":")로 제한된다. 포트 번호는 선택 사양이며 호스트네임으로부터 왼쪽은 ":", 오른쪽은 단일 슬래쉬 "/"로 제한된다. 네 번째 요소는 실제 파일 이름이나 프로그램 이름(이 경우에는, table.html)으로, 단일 슬래쉬를 더 이용하여 통상적인 방식으로 디렉토리나 서브 디렉토리를 분리한다. 이 에에서, ".html"은 이것이 HTML 파일임을 나타내는 확장자이다.

HTML(HyperText Markup Language) : 웹 서버에 의해 사용되어 웹 클라이언트에 의해 관찰되는 문서를 생성하고 접속하는 언어이다. HTML은 하이퍼텍스트 문서를 발생시킨다.

HTTP(HyperText Transfer Protocol) : 클라이언트로부터 서버로의 모든 요청이 독립적으로 처리되는 무상태(stateless) 프로토콜이다. 서버는 이전 접속에 대한 기록이 없다. URL의 시작 부분의 "http"는 요청하는 클라이언트와 목표 서버가 지정된 자원에 관한 HTTP 프로토콜을 이용하여 통신해야 함을 나타낸다.

인터넷 브라우저 또는 웹 브라우저(Internet Browser or Web browser) : HTTP와 같은 인터넷 프로토콜을 실행하고 그 결과를 사용자의 스크린 상에 디스플레이하는 그래픽 인터페이스 도구(a graphical interface tool)이다. 브라우저는 사용자가 인터넷을 "서핑(surf)"할 때 그림같은 데스크탑(pictorial desktops), 디렉토리, 검색 도구와 함께 인터넷 여행 가이드(Internet tour guide) 역할을 할 수 있다. 이 애플리케이션에서 웹 브라우저는 월드 와이드 웹과 통신하는 클라이언트 서비스이다.

클라이언트 캐쉬(Client cache) : 전형적으로 클라이언트에 의해 액세스되는 객체의 주된 그룹(캐쉬)으로서 사용된다. WWW 환경에서, 클라이언트 캐쉬는 전형적으로 웹 브라우저에 의해 구현되고, 현재 호출(current invocation) 중에 액세스되는 캐쉬 객체, 즉 비영속(non persistent) 캐쉬일 수도 있고, 호출과 교차하는(across invocations) 객체를 캐싱할 수도 있다.

캐싱 프록시(Caching proxies) : 네트워크 내에서 클라이언트를 대신해서, 캐쉬 복사본(a cached copy)을 반환하여 객체를 배치하는 에이전트(agent) 역할을 하는 전문화된 서버이다. 캐싱 프록시는 클라이언트 캐쉬로부터의 캐쉬 실패(cache misses)의 결과로서 호출되므로 전형적으로 2 차 계층 이상의 캐쉬 역할을 한다.

META-태깅(tagging) : 주어진 객체와 정보의 집합이다. 예를 들어, HTTP에서, 정보는 HTTP 헤더의 필드 내에 있는 요청 및 응답 모두와 연합될 수 있다. 예를 들어, HTTP 클라이언트는 돌려보낸 페이지(a returned page)가 요청되는 URL을 명시할 수 있다.

월드 와이드 웹(WWW 또는 Web)의 인기가 증가함에 따라, 인터넷을 통한 트래픽도 상당히 증가해왔다. 웹은 이제 네트워크 성능의 주된 병목(bottlenecks) 중 하나가 되었다. 예컨대, 웹 사용자는 느린 네트워크 링크를 통해 서버에 접속한 사용자가 문서나 정보를 요청할 때 상당한 지연을 경험할 것이다. 웹이 네트워크를 통해 사용자에게 정보를 전송하는 것은 네트워크를 통한 트래픽의 계층을 더 증가시킨다. 증가된 트래픽은 다른 사용자의 요청에 이용할 수 있는 대역폭을 감소시켜, 추가적인 잠재 지연일 일으킨다.

클라이언트를 위한 액세스 지연을 감소시키기 위해, 종래 기술은 네트워크 노드에서 사용자에게 보다 가까운 곳에 인기있는 정보의 문서의 복사본을 저장하기나 캐싱(cache)해서, 액세스 지연을 보다 수용가능하게 한다. 캐싱은 네트워크 상의 다양한 지점에서 구현될 수 있다. 예를 들어, 큰 대학이나 회사는 자신의 지역 캐쉬를 소유할 수 있어서, 이곳으로부터 그 네트워크에 기입한 모든 사용자들이 문서를 인출할 수 있다. 지역 캐쉬는 전문 서버 형태로 구현될 수 있다.

전문 서비는, 캐싱 프록시라 불리는데, 클라이언트를 대신해서 요청된 정보의 임의의 잠재적인 캐싱 복사본을 배치하는 에이전트 역할을 할 수 있다. 캐싱 프록시는 실패한 클라이언트 캐쉬로부터의 남겨진 실패(misses)와만 관련되므로 보통 2 차 계층 이상의 캐쉬 역할을 한다. 클라이언트 캐쉬는 다양한 웹 브라우저로 구축된다. 클라이언트 캐쉬는 클라이언트 호출 중에 액세스된 문서(모자이크(Mosaic)와 같은 비영속 캐쉬)만을 저장할 수도 있고, 호출과 교차하여 문서를 캐싱할 수도 있다.

웹 문서를 캐싱하는 이전의 작업은 온-디맨드 캐싱(on-demand caching) 영역에 주로 존재해왔다. 온 디맨드 캐싱은 객체가 요청된 이후에 캐싱이 결정되는 것을 의미한다. 이를테면, 에이브람즈(Abrams) 등의 "Caching Proxies : Limitations and Potentials", Proc. 4th International World Wide Web Conference(1996)는 웹을 위해 조사된 서로 다른 온 디맨드 캐싱 방침을 설명하고 있다.

인터넷 서비스 제공자(Internet Service Provider : ISP) 네트워크나 협력 네트워크에서, 프록시 계층 내의 서로 다른 지역 프록시나 부문 프록시(departmental proxies)는 서로 다른 조회 습성을 관찰하거나 서로 다른 네트워크 트래픽과 지연을 직면할 수 있다. 프록시 계층의 존재는 모든 중간 프록시, 클라이언트, 이들의 부분집합이 사전 인출(prefecthch)을 수행할 지여 여부에 대한 사전 인출 결정을 더 복잡하게 한다. 따라서, 프록시 서버 계층과 관련 클라이언트에 대해 효과적으로 사전 인출 활동을 조정할 수 있는 사전 인출 방법 및 시스템이 필요하다.

따라서, 본 발명은 컨텐트 서버(a content server)로부터 선택 정보(select information)를 사전 인출(prefetch)하여 통신 네트워크와 하나 이상의 프록시 서버를 통해 컨텐트 서버와 연관된 하나 이상의 클라

35-3

이언트에 대해 선택 정보가 용이하게 액세스될 수 있도록 하는 방법을 제공하는데, 이 방법은 프록시 서버 중 적어도 하나로부터 정보를 요청하는 단계와, 클라이언트 중 어느 하나에 의한 이전 인출 또는 요청된 정보로의 액세스에 기초하여 클라이언트를 위한 사전 인출 힌트 정보(prefetch hint information)를 설정하는 단계와, 사전 인출 힌트 정보를 이용하여 요청된 정보에 주석(annotate)을 다는 단계와, 사전 인출 힌트 정보 및 요청 정보에 기초하여 서버 중 어느 하나로부터 선택 정보를 사전 인출하는 단계를 포함한다.

전형적으로, 요청된 정보는 요청된 객체를 포함하고, 선택 정보는 인터넷 데이터와 같은 사전 인출 후보 객체를 포함한다. 이 경우, 요청된 객체는 인터넷 데이터와 연관된 HTTP 링크를 통해 선택 객체를 조회할 수 있는데, 프록시 서버는 요청된 객체의 객체 헤더 내에 저장된 요청된 객체 URL의 HTTP 프로토콜을 통해 요청된 객체에 대한 선택 객체의 관계를 식별한다.

바람직하게, 사전 인출 힌트 정보 설정 단계는 클라이언트와 관련된 데이터 조회 패턴(data reference patterns)을 추적하는 단계와, 요청된 객체와 사전 인출 후보 객체 사이의 관계에 따라 데이터 조회 패턴을 객체 식별자 그룹으로 조직하는 단계와, 상기 그룹 중 어느 것이 사전 인출 후보 객체와 요청된 객체 사이에서 선택 정보로서 사전 인출 후보 객체의 사전 인출을 보장하기에 충분한 관계를 갖는지를 결정하는 단계를 포함한다. 따라서, 상기 클라이언트의 데이터 조회 패턴은 요청된 정보의 과거의 인출이나 이전 액세스에 기초하는 것이 유리한데, 데이터 조회 패턴은 상기 프록시 서버 중 하나와 연관된 상기 클라이언트 중 다른 하나에 의한 과거의 인출이나 액세스 내에 존재하는 임의의 공통적인 경향을 나타내도록 선택된다. 이는 사전 인출 힌트 정보를 정의하는 데 사용될 수 있다.

데이터 조회 패턴 추적 및 조직 단계는 조회 데이터 패턴을 프록시 서버에 저장하는 데이터베이스를 설정하는 단계와, 각각의 데이터베이스 내에 필드를 배열하되, 필드는 객체 식별자와 객체 카운트(object counts)를 포함하고, 객체 식별자는 요청된 객체를 서로 구별하며, 객체 카운트는 요청된 객체가 상기 서버 중 하나에 의해 요청된 회수를 나타내는 단계와, 데이터베이스와 액세스 리스트(access lists)를 결합하되, 각각의 액세스 리스트는 각각의 요청된 객체와 연관된 하나 이상의 사전 인출 후보 객체를 포함하는 단계를 포함한다.

이러한 데이터베이스는 요청된 객체와 사전 인출 후보 객체 사이의 관계를 규정하고 요청된 객체와 사전 인출 후보 객체에 관한 액세스 통계를 추적하는 데 사용되는데, 액세스 통계는 요청된 객체의 부모 객체(a parent-object) 액세스 빈도와 사전 인출 후보 객체의 자녀 객체(a childOobject) 액세스 빈도를 포함하고, 부모 객체 액세스 빈도는 요청된 객체 중 특정한 하나가 상기 클라이언트 중 적어도 하나에 의해 요청된 회수를 나타내고, 자녀 객체 액세스 빈도는 특정한 하나의 액세스 바로 다음에 사전 인출 후보 객체 중 하나 이상이 실제로 요청된 회수를 나타낸다. 이로 인해, 자녀 객체 액세스 빈도와 연관된 상기 부모 객체 액세스 빈도에 대한 상기 자녀 객체 액세스 빈도의 비율에 기초하여 사전 인출 후보 객체의 사전 인출 리스트를 선택할 수 있는데, 사전 인출 리스트는 참조 확률을 초과하는 비율을 갖는 사전 인출 후보 객체만을 포함하게 된다.

일반적으로, 사전 인출 후보 객체의 서브셋만이 실제로 사전 인출되고 상기 프록시 서버 중 하나 또는 상기 클라이언트 중 하나에서 수신되는데, 전형적으로 이러한 사전 인출 후보 객체를 선택 정보로서 선택하는 것은 상기 프록시 서버 중 하나에서 이용가능한 저장 크리 및/또는 상기 프록시 서버 중 하나에서의 액세스 시간과 같은 인자에 기초한다. 따라서, 사전 인출 후보 객체는 소정의 대응하는 사전 인출 값과 연관된 우선 순위에 따라 사전 인출된다.

바람직한 실시에에서, 요청된 정보에 주석을 다는 단계는 인터넷 컨텐트 선택을 위한 플랫폼(a Platform for Internet Content Selection : PICS) 포맷, 또는 PICS 포맷과 호환가능한 라벨 프로토콜(a label protocol)을 갖는 사전 인출 라벨 내에 상기 사전 인출 힌트 정보를 배치하는 단계를 포함한다. 이러한 PICS 포맷의 구체적인 구현에 있어서, 객체 식별자 카테고리와 조회 식별자 카테고리(referrer identifier category)가 정의되고, 상기 방법은 상기 조회 식별자 카테고리를 상기 컨텐트 서버로 돌려보내, 실제로 조회된 임의의 사전 인출 후보 객체를 식별하는 단계를 더 포함한다. 인터넷 컨텐트 선택 프로토콜이나 PICS 호환가능 프로토콜을 위한 플랫폼에 따라 사전 인출 힌트 정보를 조직하는 것은 이미 많은 브라우저에 의해 지원되는 포맷을 이용한다.

전형적으로 상기 프록시 서버에는 계층이 존재한다. 송신 노드(a sending node)와 수신 노드(a receiving node)가 정의되는데, 사전 인출 힌트 정보는 초기 상태에서 최종 상태로 수정하는 것이 바람직하고, 초기 상태는 송신 노드에 의해, 최종 상태는 상기 수신 노드에 의해 결정된다. (사전 인출 선택 정보가 수신 노드에 의해 개시된다고 가정하면) 상기 최종 상태의 사전 인출 힌트 정보는 궁극적인 사전 인출 결정을 위한 기초로서 형성된다.

달리 말해서, 사전 인출 힌트 정보는 계층을 통해 프록시 서버로부터 클라이언트에게 전달될 때 해석되고 갱신될 수 있다. 바람직하게, 프록시 서버는 요청된 정보에 대한 지역 조회(local references)가 존재하고 상기 지역 계층 서버의 카운트가 최소 참조 카운트를 초과하면(즉, 지역적으로 믿을 수 있는 통계를 이용할 수 있으면), 상기 사전 인출 힌트 정보를 갱신한다. 또한, 사전 인출 힌트 정보는 프록시 계층 내의 캐쉬 상태(예를 들어, 계층 내의 어느 곳에서 특정한 사전 인출 후보 객체가 캐쉬될 수 있는지를 나타냄)에 기초하여 갱신될 수 있다.

본 발명은 또한 네트워크(25)를 통해 컨텐트 서버(20)로부터 선택 정보를 사전 인출하는 데이터 처리 시스템을 제공하는데, 상기 시스템은 네트워크와 연관된 하나 이상의 프록시 서버(24), 컨텐트 서버와 하나 이상의 상기 프록시 서버와 상기 네트워크를 통해 통신하는 클라이언트를 포함하며, 상기 시스템은 프록시 서버 중 적어도 하나로부터 정보를 요청하는 수단과, 클라이언트 중 어느 하나에 의한 이전 인출 또는 요청된 정보로의 액세스에 기초하여 클라이언트를 위한 사전 인출 힌트 정보를 설정하는 수단과, 사전 인출 힌트 정보를 이용하여 요청된 정보에 주석을 다는 수단과, 사전 인출 힌트 정보 및 요청 정보에 기초하여 서버 중 어느 하나로부터 선택 정보를 사전 인출하는 수단을 포함한다.

바람직한 실시에에서, 요청된 정보는 요청된 객체를 포함하며, 선택 정보는 사전 인출 후보 객체를 포함하고, 상기 각각의 서버와 클라이언트 중 적어도 하나는 주된 캐쉬와 사전 인출 캐쉬 사이에 적어도 논리적

구별(logical separation)을 갖는 메모리를 구비하며, 상기 사전 인출 선택 정보는 상기 사전 인출 캐쉬에 저장된다.

사전 인출 수단은 객체를 사전 인출하고 사전 인출된 객체를 우선 순위 방안(a priority scheme)에 기초하여 사전 인출 캐쉬에 저장하되, 우선 순위 방안은 최고 사전 인출 값을 갖는 객체를 사전 인출 캐쉬의 이용가능한 메모리에 대해 먼저 저장하고, 사전 인출 값은 하나 이상의 클라이언트에 의한 사전 인출 객체에 대한 개별적이거나 통합된 조회 가능성(reference probability)에 기초하는 것이 바람직하다. 특히, 사전 인출 값은 사전 인출 힌트 정보 리스트 상의 사전 인출 후보 객체가 클라이언트 중 특정한 하나에 의해 액세스될 추정된 액세스 확률과 사전 인출 후보 객체 중 선택된 하나를 저장하는 데 필요한 저장 크기에 기초한 객체의 액세스 비용에 기초할 수 있다.

바람직한 실시예에서, 사전 인출 수단은 사전 인출 값을 계산하는 계산 수단과 계산 수단에 의해 계산된 사전 인출 값을 평가하는 평가 수단을 포함하는 사전 인출 처리기를 포함하고, 계산 수단은 사전 인출 힌트 정보 리스트 상의 각각의 사전 인출 객체 식별자에 대해 사전 인출 값을 계산하고, 평가 수단은 다수의 후보 사전 인출 객체의 사전 인출 값을 평가하고 사전 인출 힌트 정보 리스트 상의 최고 사전 인출 값과 연관된 후보 사전 인출 객체의 캐쉬의 우선 순위를 매긴다. 사전 인출 수단은 프록시 서버 중 적어도 하나 안에 제 1 사전 인출 처리기를, 클라이언트 중 적어도 하나 안에 제 2 사전 인출 처리기를 포함한다(즉, 사전 인출은 계층 내의 서로 다른/다수의 장소에서 수행될 수 있다).

또한, 바람직한 실시예는 클라이언트 중 하나의 메모리에 저장된 클라이언트 객체 요청 처리기를 포함하되, 클라이언트 객체 요청 처리기는 프록시 서버나 네트워크로부터 요청된 객체를 요청하기 이전에 요청된 객체를 위한 주된 캐쉬(main cache)나 사전 인출 캐쉬를 검색하고, 클라이언트 객체 요청 처리기는 객체가 사전 인출 캐쉬 내에 있는 경우, 사전 인출 객체가 상기 클라이언트에 의해 실제로 액세스되었음을 나타내는 사전 인출 조회 메시지(a prefetch referenced message)를 서버로 발생시킴으로써 컨텐트 서버나 프록시 서버로 통지한다. 마찬가지로, 프록시 서버 중 적어도 하나는 요청된 객체가 사전 인출 캐쉬 내에 위치하는 경우 캐쉬 관리자 루틴(a cache manager routine)을 호출한 후 사전 인출 조회 메시지를 전달하는 사전 인출 조회 메시지 처리기를 포함하는 것이 바람직하다. 이는 서버가 자녀 객체에 직접 접속되지 않더라도 클라이언트나 프록시 서버 내에 있는 사전 인출 캐쉬로부터 정확한 액세스 기록을 이용할 수 있기 때문에, 서버가 정확한 액세스 기록을 유지할 수 있도록 보장한다.

본 발명은 이러한 데이터 처리 시스템에서 사용되는 컨텐트 또는 프록시 서버를 제공하는데, 요청된 정보는 요청된 객체를 포함하고, 선택 정보는 사전 인출 후보 객체를 포함하며, 상기 서버는 부모 객체 식별자와 연관된 자녀 객체 식별자를 저장하는 통계 테이블과, 통계 테이블 내의 정보에 기초하여 사전 인출 힌트 정보 리스트를 발생시키는 발생 수단을 포함하되, 통계 테이블은 부모 객체의 제 1 카운트를 포함하고, 제 1 카운트는 부모 객체가 클라이언트에 의해 조회된 빈도를 나타내며, 통계 테이블은 자녀 객체의 제 2 카운트를 포함하고, 제 2 카운트는 대응 자녀 객체가 부모 객체 중 선택된 하나가 조회된 후 조회되는 빈도를 나타낸다.

일 실시예에서, 발생 수단은 통계 테이블로부터 제 1 카운트, 제 2 카운트, 자녀 객체 식별자, 부모 객체 식별자의 입력을 수용하고, 발생 수단은 부모 객체와 그들의 연관된 자녀 객체 사이의 액세스 관계를 평가하는 평가 수단을 포함하되, 평가 수단은 자녀 객체가 대응 부모 객체에 대한 클라이언트에 의한 조회 이전에, 바로 다음 또는 후속적으로 적절한 빈도로 조회되는 경우, 사전 인출 힌트 정보 리스트 상에 자녀 객체 식별자를 배치한다. 적절한 빈도는 대응 부모 액세스에 대한 자녀 액세스의 비율에 기초할 수 있다. 전형적으로, 이는 상기 부모 객체 식별자 각각에 대해 제 1 카운트에 대한 제 2 카운트의 비율을 계산하고, 비율을 힌트 참조값과 비교하며, 비율이 힌트 참조값을 초과하는 경우 사전 인출 힌트 정보 상의 자녀 객체와 연관된 부모 객체를 그룹화하거나 연합하는 것과 관련된다.

바람직하게, 서버는 실제로 요청되는 자녀 객체에 기초하여 통계 테이블을 갱신하는 통계 갱신 수단을 더 포함하되, 통계 갱신 수단은 부모 객체 중 조회하는 하나(a referring one)의 신원을 먼저 검색하되, 조회하는 하나, 즉 조회자 객체(referrer object)는 자녀 객체가 실제로 요청되기 이전에 조회되며, 자녀 객체는 자신의 대응 제 2 카운트를 상기 통계 테이블 내에서 갱신시키면서 실제로 요청된다. 통계 테이블은 사전 인출 조회 메시지 처리기로부터의 사전 인출 조회 메시지 수신 시 갱신되어야 한다.

서버는 프록시 서버이고, 사전 인출 힌트 정보 리스트를 수신하고 갱신하는 사전 인출 갱신 수단을 더 포함하는 것이 바람직하다. 사전 인출 갱신 수단은 제 1 지역 참조 카운트와 제 1 카운트를 비교하고, 사전 인출 갱신 수단은 제 1 카운트가 제 1 지역 참조 카운트를 초과하는 자녀 객체 식별자 각각에 대해 발생 수단을 호출하며, 사전 인출 갱신 수단은 제 1 카운트가 제 1 지역 참조 카운트를 초과하지 않는 경우, 원래 사전 인출 힌트 정보 리스트를 유지한다. 보다 정교한 접근 방법에서, 사전 인출 갱신 수단은 제 1 카운트를 제 1 및 제 2 지역 참조 카운트와 비교하여, (1) 제 1 카운트가 제 1 지역 참조 카운트를 초과하지 않으면 원래 사전 인출 힌트 정보 리스트를 유지하고, (2) 제 1 카운트가 제 1 지역 참조 카운트를 초과하면서 제 2 지역 참조 카운트를 초과하지 않으면 원래 사전 인출 힌트 정보 리스트를 수정하며, (3) 제 1 카운트가 제 1 및 제 2 지역 참조 카운트를 초과하면 원래 사전 인출 힌트 정보 리스트를 교체한다.

바람직하게, 서버는 요청된 객체를 요청한 하위 계층의 프록시 서버나 클라이언트에게로 요청된 객체를 돌려보내는 프록시 객체 처리기를 더 포함하되, 프록시 객체 처리기는 요청된 객체가 사전 인출 객체가 아닌 경우 요청된 객체를 돌려보내고 따라서 통계 테이블을 갱신한다. 따라서, 프록시 객체 처리기는 사전 인출 캐쉬로부터 요청된 객체를 요청 클라이언트나 프록시 서버로 돌려보내고, 객체가 사전 인출 요청에 따라 요청된 경우 상기 사전 인출 캐쉬로부터 삭제가능한 것으로 객체를 표시한다. 프록시 객체 처리기는 요청된 객체가 주된 캐쉬나 사전 인출 캐쉬 내에 있지 않으면 객체 요청을 다른 서버로 보낸다. 프록시 객체 처리기는 요청된 객체가 사전 인출 요청과 연관되지 않고 요청된 객체가 사전 인출 캐쉬나 주된 캐쉬 내에 위치하면 통계 테이블을 갱신한다.

다른 측면에서 보면, 본 발명은 컨텐트 서버로부터 선택 객체를 사전 인출하여 통신 네트워크를 통해 하나 이상의 프록시 서버와 연관된 클라이언트를 위해 선택 객체를 용이하게 이용할 수 있도록 만드는 방법을 제공하되, 이 방법은 프록시 서버 중 적어도 하나로부터 요청 정보를 요청하는 단계와, 클라이언트에 의한

이전 인출 또는 요청된 정보로의 액세스에 기초하여 대응 클라이언트의 데이터 조회 패턴을 식별하는 단계와, 데이터 조회 패턴으로부터 사전 인출 힌트 정보 리스트로 사전 인출 힌트 정보를 발전시키는 단계와, 사전 인출 힌트 정보를 이용하여 요청된 객체에 주석을 다는 단계와, 사전 인출 힌트 정보와 현재 요청된 객체에 기초하여 상기 서버 중 적어도 하나로부터 선택 객체를 사전 인출하는 단계를 포함한다.

따라서, 사전 인출된 선택 정보는 사전 인출 후보 객체를 포함하고, 사전 인출 힌트 정보를 설정하는 단계는 상기 프록시 서버의 계층 내의 송신 노드에서 사전 인출 힌트 정보를 설정하는 단계를 더 포함하되, 사전 인출 후보 객체는 상기 프록시 서버의 계층을 통과하며, 선택 정보는 사전 인출 힌트 정보에 기초하여 계층 내의 수신 노드에서 사전 인출된다.

전형적으로 네트워크와 연관된 프록시 서버 계층이 존재하는데, 프록시 서버 계층은 다른 프록시 서버보다 클라이언트와 보다 긴밀하게 연관된 하위 계층을 포함한다. 바람직한 일 실시예에서, 사전 인출 힌트 정보는 요청된 정보에 대한 지역 조회가 존재하고 지역 참조 서버의 카운트가 최소 참조값을 초과하면 프록시 서버 계층을 통해 갱신되고, 이와는 달리(또는 추가적으로) 사전 인출 힌트 정보는 상위 계층에 있는 개체 상태를 분석함으로써 갱신될 수도 있는데, 계층의 최상위 계층은 네트워크와 가장 긴밀히 연관되어 있다.

다른 측면에서 볼 때, 본 발명은 네트워크를 따라 서버로부터 클라이언트로의 데이터 흐름을 개선하는 처리 시스템을 제공하는데, 이 처리 시스템은 네트워크와 연관되는 프록시 서버와, 네트워크와 연관되는 컨텐트 서버와, 상기 프록시 서버 중 하나 이상과 통신하도록 구성된 클라이언트와, 상기 프록시 서버 중 적어도 하나에 저장되며, 부모 객체 식별자와 연관된 자녀 객체 식별자, 부모 객체의 제 1 카운트, 자녀 객체의 제 2 카운트를 저장하되, 제 1 카운트는 부모 객체가 클라이언트에 의해 조회되는 회수를 나타내고, 제 2 카운트는 대응 자녀 객체가 상기 부모 객체 중 하나가 조회된 후 조회되는 회수를 나타내는 통계 테이블과, 제 1 카운트와 대응 제2 카운트의 입력을 수용하여 사전 인출 힌트 정보 리스트를 발생시켜, 사전 인출 정보 리스트를 유도하는 발생 수단을 포함하되, 상기 발생 수단은 상기 부모 객체 식별자 각각에 대해 제 1 카운트에 대한 제 2 카운트의 비율을 계산하는 계산 수단과, 힌트 참조값과 비율을 비교하고, 비율이 힌트 참조값을 초과하면 사전 인출 힌트 정보 상의 연관된 부모 객체와 자녀 객체를 그룹화하거나 연합시키는 수단과, 사전 인출 힌트 정보와 클라이언트에 의해 실제로 요청된 부모 객체에 기초하여 자녀 객체를 사전 인출하는 사전 인출 수단을 포함한다.

바람직한 실시예에서, 상기 프록시 서버 각각은 제 1 프로세서, 제 1 메모리, 데이터를 저장하는 제 1 데이터 저장 수단, 제 1 주된 캐쉬와 제 1 사전 인출 캐쉬 사이에 적어도 논리적인 구별을 갖는 제 1 메모리, 데이터를 저장하는 제 2 저장 수단을 구비하고, 적어도 하나의 클라이언트는 제 3 프로세서, 제 3 메모리, 데이터를 저장하는 제 3 저장 수단을 구비하되, 제 3 메모리는 제 2 주된 캐쉬와 제 2 사전 인출 캐쉬 사이에 적어도 논리적인 구별을 갖고, 사전 인출 처리 수단은 제 1 사전 인출 캐쉬나 제 2 사전 인출 캐쉬 내에 자녀 객체를 저장한다.

바람직한 실시예에서, 사전 인출 수단은 사전 인출 값을 계산하는 계산 수단과, 계산 수단에 의해 계산된 사전 인출 값을 평가하는 평가 수단을 포함하고, 상기 계산 수단은 사전 인출 힌트 정보 리스트 상에 있는 각각의 자녀 객체 식별자에 대해 사전 인출 값을 계산하고 최고 사전 인출 값을 갖는 자녀 객체의 캐쉬을 사전 인출 힌트 정보 리스트 상에 우선 순위를 매기며, 사전 인출 처리 수단은 사전 인출 캐쉬 내의 이용 가능한 공간을 관리한다.

바람직하게, 시스템은 사전 인출 힌트 정보 리스트를 갱신하는 사전 인출 갱신 수단을 더 포함하는데, 사전 인출 갱신 수단은 제 1 카운트를 제 1 지역 참조 카운트 및 제 2 지역 참조 카운트와 비교하고, 제 1 지역 참조 카운트는 한 서버로부터의 통계에 기초하며, 제 2 지역 참조 카운트는 다른 서버로부터의 통계에 기초하고, 상기 사전 인출 갱신 수단은 제 1 카운트가 제 1 지역 참조 카운트와 제 2 지역 참조 카운트를 초과하는 경우의 자녀 객체 식별자 각각에 대해 발생 수단을 호출하며, 사전 인출 갱신 수단은 제 1 카운트가 제 1 지역 참조 카운트를 초과하지 않으면 원래 사전 인출 힌트 정보 리스트를 유지한다.

이러한 시스템은 통계 테이블을 갱신하는 통계 갱신 수단과, 상기 프록시 서버 중 하나에 저장되며, 사전 인출 힌트 정보 리스트 갱신 루틴을 호출하고, 하위 계층 프록시 서버가 요청된 객체를 요청하면 하위 계층 프록시 서버로 요청된 객체를 반환하며, 요청된 객체가 사전 인출 객체가 아니면 요청된 객체를 반환하고 통계 갱신 수단을 호출하는 프록시 객체 처리기와, 상기 프록시 서버 중 하나와 연관되는 사전 인출 캐쉬를 관리하되, 프록시 객체 처리기는 하위 계층 서버가 요청된 객체를 요청하지 않을 때 호출하는 사전 인출 캐쉬 관리자와, 상기 프록시 서버 중 하나에 저장되는 프록시 객체 요청 처리기를 포함하는 것이 바람직하되, 프록시 객체 요청 처리기는 객체가 사전 인출 캐쉬 내에 존재하고 객체가 사전 인출 요청에 따라 요청되면 객체를 상기 클라이언트 중 요청하는 하나로 반환하고 그 객체에 사전 인출 캐쉬로부터 삭제 가능하다는 표시를 하며, 프록시 객체 요청 처리기는 요청된 객체가 주된 캐쉬 또는 사전 인출 캐쉬 내에 있지 않으면 객체 요청을 다른 서버로 전달하며, 프록시 객체 요청 처리기는 요청된 객체가 사전 인출 요청과 연관되지 않고 요청된 객체가 사전 인출 캐쉬나 주된 캐쉬 내에 위치하면 통계 갱신 수단과 통신한다.

또한, 이 시스템은 통계 데이블을 갱신하는 통계 갱신 수단과, 상기 프록시 서버 중 하나에 저장되는 사전 인출 조회 메시지 처리기를 포함하는 것이 바람직하되, 사전 인출 조회 메시지 처리기는 통계 갱신 수단을 호출하고, 요청된 객체가 사전 인출 캐쉬 내에 존재하면 캐쉬 관리자 루틴을 호출한 후 사전 인출 조회 메시지를 전달한다.

또한, 본 발명은 네트워크를 따라 서버로부터 클라이언트로의 데이터 흐름을 개선하는 처리 시스템을 제공하는데, 이 처리 시스템은 네트워크와 연관되는 프록시 서버와, 네트워크와 연관되는 컨텐트 서버와, 상기 프록시 서버 중 하나 이상과 통신하도록 구성된 클라이언트와, 상기 컨텐트 서버와 프록시 서버 중 적어도 하나에 저장되며, 부모 객체 식별자와 연관된 자녀 객체 식별자, 부모 객체의 제 1 카운트, 자녀 객체의 제 2 카운트를 저장하되, 제 1 카운트는 부모 객체가 클라이언트에 의해 조회되는 회수를 나타내고, 제 2 카운트는 대응 자녀 객체가 상기 부모 객체 중 선택된 하나가 조회된 후 바로 다음에 일시적으로 조회되는 회수를 나타내는 통계 테이블과, 통계 테이블로부터 제 1 카운트, 제 2 카운트, 자녀 객체 식별자, 부모

객체 식별자를 입력으로 수용하며, 부모 객체와 그들의 연관된 자녀 객에 사이의 관계를 평가하는 평가 수단을 포함하되, 평가 수단은 자녀 객체가 대응 부모 객체에 대한 클라이언트의 조회 이전에 바로 다음에 후속하여 적절한 빈도로 조회된 경우 자녀 객체 식별자를 사전 인출 힌트 정보 리스트 상에 배치하는 발생 수단과, 사전 인출 힌트 정보와 클라이언트에 의해 실제로 요청된 부모 객체에 기초하여 자녀 객체를 사전 인출하는 사전 인출 수단을 포함한다.

바람직한 실시예에서, 사전 인출 수단은 사전 인출 힌트 정보 리스트 상에서 식별된 자녀 객체에 저장 우선 순위를 할당하는 할당 수단과, 저장 우선 순위에 따라 자녀 객체 중 선택된 하나를 저장하는 선택적 저장 수단을 더 포함한다. 할당 수단은 사전 인출 힌트 정보 리스트 상의 자녀 객체가 클라이언트 중 특정한 하나에 의해 액세스될 추정 액세스 확률 및 자녀 객체 중 선택된 하나를 저장하는 데 필요한 저장 크기에 기초한 자녀 객체의 액세스 비용에 기초하여 저장 우선 순위를 할당한다.

따라서, 정보는 프록시 계층 내에 사전 인출되어 클라이언트의 네트워크(즉, 인터넷)를 통한 액세스 시간을 감소시킨다. 선택 정보는 컨텐트 서버로부터 사전 인출되어 선택 정보를 네트워크 내의 컨텐트 서버와 연관된 클라이언트가 용이하게 액세스할 수 있도록 만든다. 클라이언트는 네트워크 내의 프록시 서버인 것이 바람직하다. 클라이언트는 프록시 서버 중 적이도 하나로부터 정보를 요청한다. 컨텐트 서버는 클라이언트에 의해 요청된 정보의 과거 데이터 액세스 패턴에 기초하여 클라이언트를 위한 사전 인출 힌트 정보를 발생시킨다. 컨텐트 서버는 사전 인출 힌트 정보를 이용하여 요청된 정보에 주석을 단다. 하나 이상의 프록시 서버는 동적 사용 통계에 기초하여 사전 인출 힌트 정보를 동적으로 갱신할 수 있다. 사전 인출 힌트 정보는 객체가 프록시 계층을 통과할 때 갱신되어 수행된 사전 인출 동작, 프록시 계층의 상위 계층에서의 캐쉬 상태, 지역 조회 패턴과 같은 기타 지역 고려 사항을 반영한다. 선택 정보는 사전 인출 힌트 정보와 요청된 정보에 기초하여 컨텐트 서버나 프록시 서버 중 임의의 하나로부터 사전 인출될 수 있다. 사전 인출 힌트 정보는 조회 액세스 확률과 저장 고려 사항에 기초하여 사전 인출 값을 할당받을 수 있다. 사전 인출 힌트 정보 리스트 내에서 최고 사전 인출 값을 갖는 선택 정보는 사전 인출 캐쉬가 가득 찰 때까지 가장 먼저 그리고 잠재적으로 캐쉬된다.

도면의 간단한 설명

도 1은 컨텐트 서버(content server)(즉, 웹 사이트(Web sites), 네트워크(즉, 인터넷), 프록시 서버 계층, 클라이언트를 나타내는 블럭도,

도 2는 도 1보다 상세한 컨텐트 서버, 클라이언트, 프록시 서버의 블럭도,

도 3a는 프록시 서버 계층을 통해 컨텐트 서버로부터 데이터를 동적으로 사전 인출하는 방법을 나타내는 흐름도,

도 3b는 통계 테이블의 예를 도시하는 도면,

도 4는 컨텐트 서버의 동작을 나타내는 흐름도,

도 5는 통계 갱신을 도시하는 흐름도,

도 6은 사전 인출 힌트 정보의 발생을 나타내는 흐름도,

도 7은 클라이언트 동작을 나타내는 흐름도,

도 8은 사전 일출을 나타내는 흐름도,

도 9는 클라이언트 객체 처리기의 동작을 나타내는 흐름도,

도 10은 프록시 서버의 동작을 나타내는 흐름도,

도 11은 프록시 객체 요청 처리기의 동작을 나타내는 흐름도,

도 12는 사전 인출 조회 메시지 처리기(a prefetch referenced message handler)의 동작을 나타내는 흐름도,

도 13은 프록시 객체 처리기의 동작을 나타내는 흐름도,

도 14는 사전 인출 힌트 정보 갱신 루틴의 동작을 나타내는 흐름도.

바람직한 실시예의 설명

이제 첨부하는 도면을 조회하여 예에 의해 본 발명의 바람직한 실시예를 상세히 설명할 것이다.

도 1은 클라이언트(22), 프록시 서버 계층(24), 네트워크(25), 컨텐트 서버(20)를 포함하는 데이터 프로세싱 시스템을 도시하고 있다.

클라이언트는 부모 객체, 자녀 객체, 요청된 객체, 선택 객체, 페이지, 요청된 정보, 선택 정보의 형태로 정보를 요청할 수 있다. 부모 객체가 하나 이상의 연관된 자녀 객체에 대한 후속 요청을 역사적으로 처리하도록 부모 객체는 자여 객체와 연관된다. 요청된 객체는 부모 객체와 유사하지만, 선택 객체는 자녀 객체와 유사하다. 요청된 객체는 하나 이상의 객체에 의해 실제로 요청된 객체이다. 한편, 선택 객체는 선택 객체가 요청된 객체 이후에 바로 다음 일시적으로 요청될 수 있도록 요청된 객체와 그룹화되거나 연관되는 객체이다. 요청된 정보는 요청된 객체, 부모 객체, 기타 클라이언트에 의해 요청된 데이터 형태를 포함한다. 선택 정보는 자녀 객체, 선택 객체, 기타 사전 인출될 수 있는 데이터 형태를 포함한다.

클라이언트(22)는 컨텐트 서버(20) 또는 프록시 서버(24)로부터 정보를 요청할 수 있다. 그런, 프록시 서버(24)는 보통 네트워크(25)와 컨텐트 서버(20)에 대해 클라이언트(22)를 위한 매개 역할을 한다. 유리하게도, 프록시 서버(24)는 프록시 서버 계층이 요청된 정보를 포함하거나, 요청된 작업을 완료할 능력이 있는 경우에는 컨텐트 서버(20)에 접속하지 않고 클라이언트 정보 요청을 완수할 수 있다. 도 1의 데이터

처리 시스템은 인터넷에 적용될 수 있다. 예를 들어, 컨텐트 서버(20)는 웹 사이트를 포함할 수 있고, 네트워크(25)는 인터넷을 포함할 수 있다.

도 1은 데이터 처리 시스템의 구성 요소 사이에 있는 물리적 접속과 대조되는 논리적 접속(26)을 도시하고 있다. 논리적 접속(26)은 데이터 처리 시스템 내의 객체와 요청의 정보 흐름을 나타낸다. 논리적 접속(26)은 전형적으로 프록시 서버(24) 사이의 물리적 접속을 나타내지는 않는다. 논리적 접속(26)은 부하(workload) 및/또는 노드 또는 통신 링크 장애와 같은 물리적 사건으로 인해 변경될 수 있다. 서로 다른 유형의 객체는 서로 다른 전송의 논리적 경로를 따라 클라이언트(22)에게 도달한다.

사실상, 임의 개수의 클라이언트(22)와 프록시 서버(24)를 실제로 사용할 수 있지만, 도 1의 실시예는 4개의 클라이언트(22)와 4 계의 프록시 서버(24) 계층을 나타낸다. 실제로, 프록시 서버 계층 내에는 어떠한 수의 계층도 허용되며, 클라이언트(22)는 프록시 서버 계층의 어떠한 계층에도 논리적으로 접속될 수 있다. 예를 들어, 프록시 서버(24)의 계층은 단일 프록시 서버일 수도 있다. 본 발명은 단일 클라이언트(22)와 단일 프록시 서버(24)만을 이용하여 실시될 수 있다.

도시한 바와 같이, 최고 계층 프록시 서버는 계층-0 프록시 서버(30)로 지정된다. "계층-0"은 도 1의 실시예에서 다른 프록시 서버에 대한 한 프록시 서버의 상태를 정의하는 임의의 지정일 뿐이다. 최고 계층 프록시 서버는 네트워크(25)에 논리적으로 접속될 수 있다. 상위 계층 프록시 서버(21)는 하위 계층 프록시 서버(23)보다 논리적으로 긴밀하게 네트워크(25)와 연관된다. (따라서, 이 용어 중에서 계층-0 프록시 서버는 최상위 프록시 계층이고, 계층-0은 계층-1보다 "상위"이며, 계층-1은 계층-2보다 "상위"인 것을 나타냄을 이해할 것이다.)

제 1 계층-1 프록시 서버(35)와 제 2 계층-1 프록시 서버(37)는 계층-0 프록시 서버(30)에 논리적으로 접속되어 있다. 제 1 계층-1 프록시 서버(35)는 제 1 계층-2 프록시 서버(40)와 제 2 계층-2 프록시 서버(43)에 접속되이 있다 제 1 계층-2 프록시 서버(40)는 제 1 계층-3 프록시 서버(50)와 제 2 계층-3 프록시 서버(55)에 논리적으로 접속되어 있다.

클라이언트(22)는 제 1 클라이언트(600), 제 2 클라이언트(603), 제 3 클라이언트(700), 제 4 클라이언트(703)를 포함한다. 제 1 클라이언트(600)와 제 2 클라이언트(603)는 제 1 계층-3 프록시 서버(50)에 논리적으로 접속되어 있다. 제 3 클라이언트(700)와 제 4 클라이언트(703)는 제 2 계층-3 프록시 서버(55)에 논리적으로 접속되이 있다.

이제 데이터 처리 시스템은 제 2 클라이언트(603)의 관점에서 설명된다. 제 2 클라이언트(603)는 자신의 바로 상위 계층 프록시에 접속되어 있다 설명한 바와 같이, 바로 상위 계층 프록시는 제 1 레벨-3 프록시(50)이다. 제 2 클라이언트(603)는 제 1 계층-2 프록시 서버(40), 제 1 계층-1 프록시 서버(35), 계층-0 프록시 서버(30)에 각각 접속된다. 제 2 클라이언트(603)는 네트워크(25)를 통해 다양한 컨텐트 서버(20)를 액세스할 수 있다. 제 2 클라이언트(603)를 위한 전형적인 통신 경로는 바로 상위 계층 프록시(즉, 제 1 계층-3 프록시 서버(50))를 통해서이지만, 제 2 클라이언트(603)이나 하위 계층 프록시는 다른 상위 계층 프록시 서버(24)나 네트워크(25)와 직접 통신할 수도 있다. 제 2 클라이언트(603)는 캐쉬을 관리하기 위해 자신의 클라이언트를 가질 수도 있다.

제 2 클라이언트(603)의 관점에서, 소정의 프록시 서버는 자신의 프록시 계층의 일부가 아니다. 예를 들어, 제 2 계층-3 프록시 서버(55), 제 2 계층-2 프록시 서버(43), 제 2 계층-1 프록시 서버(37)는 제 2 클라이언트의 프록시 서버 계층의 일부가 아니다.

클라이언트(22)는 개인용 컴퓨터, 워크 스테이션, 셋 탑 박스(set top box) 등을 포함할 수 있지만, 여기에 국한되지는 않는다. 네트워크(25)는 인터넷, 월드 와이드 웹, 인트라넷, LAN(local area networks) 등을 포함할 수 있지만, 여기에 국한되지는 않는다.

프록시 서버(24)와 컨텐트 서버(20)는 상용화된 다양한 컴퓨터를 포함한다. 예를 들어, 프록시 서버(24)는 IBM 사에서 판매하는 인터넷 접속 서버(Internet Connection Server : ICS)로서 구현될 수도 있다. 마찬가지로, 컨텐트 서버(20)는 (역시 IBM 사로부터 입수가능한) 로터스 고 웹 서버(Lotus Go Web server), 로터스 도미노 서버(Lotus Domino server) 등으로서 구현될 수도 있다. 프록시 서버(24) 또는 컨텐트 서버(20)는 임의의 컴퓨팅 노드(computing node) 상에서 실행될 수 있는데, 이는 (역시 IBM 사로부터 입수가능한) S/390 SYSPLLX, SP2, RS6000 워크 스테이션과 같은 제품을 포함할 수 있으나, 여기에 국한되지는 않는다. 요컨대, 프록시 서버(24) 또는 컨텐트 서버(20)는 월드 와이드 웹 페이지 액세스, 원격 파일 전송, 전자 우편, 클라이언트(22)에 대한 트랜잭션 지원을 제공할 수 있는 임의의 범용 컴퓨터 및 관련 소프트웨어를 포함할 수 있다.

도 1의 데이터 처리 시스템은 통상적인 프록시 계층의 속성을 이용하여 데이터 전송 효율을 유지할 수 있다. 예컨대, 요청된 객체가 하위계층 프록시 서버 상에서 지역적으로 이용가능하지 않은 경우, 다음 상위 계층 프록시로 이용불가능한 객체에 대한 요청이 만들어진다. 상위 계층 프록시가 이전에 자신의 캐쉬 메모리 내에 그 객체를 캐싱했으면, 상위 계층 프록시는 하위 계층 프록시 서버를 통해 클라이언트에게 요청된 객체를 전달할 것이다. 그렇지 않으면, 상위 계층 프록시 서버는 자신의 다음 상위 계층 프록시로부터 그 객체를 얻으려고 시도할 것이다. 요청된 객체를 얻으면, 객체는 하위 계층 프록시나 그 객체를 요청한 클라이언트에게 전달된다.

도 2는 도 1에 도시한 데이터 처리 시스템의 보다 상세한 예를 도시하고 있다. 데이터 처리 시스템은 프록시 서버(24), 컨텐트 서버(20), 네트워크(25), 클라이언트(22)를 포함한다. 데이터 처리 시스템은 사전 인출 유닛(250), 통계 테이블(261), 통계 갱신 유닛(252), 주된 캐쉬(297), 사전 인출 캐쉬(298)를 포함하는 것이 바람직하다.

사전 인출 유닛(250)은 프록시 서버(24) 내에 제 1 사전 인출 처리기(248)를, 클라이언트(22) 내에 제 2 사전 인출 처리기를 포함하는 것이 바람직하다. 통계 테이블(261)은 프록시 서버(24)와 연관된 제 1 통계 테이블(293)과 컨텐트 서버와 연관된 제 2 통계 테이블(267)을 포함한다. 통계 갱신 유닛(252)은 제 1 통계 테이블(293)을 갱신하는 제 1 갱신 통계 유닛(264)과 제 2 통계 테이블(267)을 갱신하는 제 2 갱신 통

게 유닛(269)을 포함한다. 주된 캐쉬(297)는 각 프록시 서버(24) 내에 제 1 주된 캐쉬(294)를, 각 클라이언트(22) 내에 제 2 주된 캐쉬(246)를 포함한다. 사전 인출 캐쉬(298)는 프록시 서버(24) 내에 제 1 사전 인출 캐쉬(296)를, 클라이언트(22) 내에 제 2 사전 인출 캐쉬(249)를 포함한다.

프록시 서버(24)는 네트워크(25)를 통해 요청을 서빙(serve)할 수 있는 컴퓨팅 노드이다. 프록시 서버(24)는 제 1 프로세서(280), 제 1 메모리(290), 제 1 저장 유닛(281)을 포함한다. 제 1 메모리(290)는, 예를 들어, RAM(random access memory)를 포함할 수 있다. 제 1 저장 유닛(281)은, 이를테면, 직접 액세스 저장 디바이스(direct access storage device : DASD)를 포함할 수 있다. 제 1 메모리(290)는 프록시 서버 로직(295)을 저장하는데, 이는 컴퓨터 실행가능 코드로서 구현된다. 프록시 서버 로직(295)은 제 1 프로세서(280)에 의한 실행을 위해 제 1 저장 유닛(281)으로부터 제 1 메모리(290)로 로딩된다. 프록시 서버 로직(295)의 상세한 동작은 도 10에서 도시하고 있다.

프록시 서버 로직(295)은 프록시 객체 요청 처리기(283), 프록시 객체 처리기(284), 제 1 사전 인출 처리기(248), 사전 인출 조회 메시지 처리기(285), 사전 인출 갱신 수단(291), 제 1 갱신 통계 루틴(264), 제 1 통계 테이블(293), 제 1 주된 캐쉬(294), 제 1 사전 인출 캐쉬(296)를 포함한다.

프록시 객체 요청 처리기(283)의 상세한 동작은 도 11에서 도시하고 있다. 프록시 객체 처리기(284)의 상세한 동작은 도 13에서 도시하고 있다. 사전 인출 조회 메시지 처리기(285)의 동작은 도 12에서 도시하고 있다. 통계 갱신 유닛(252)(즉, 제 1 갱신 통계 루틴(264)의 상세한 동작은 도 5에서 도시하고 있다.

사전 인출 유닛(250)(즉, 제 1 사전 인출 처리기(248))의 동작은 도 8에서 상세하게 도시하고 있다. 사전 인출 갱신 유닛(291) 또는 사전 인출 힌트 정보 갱신 루틴의 동작은 도 14에서 도시하고 있다. 프록시 서버 로직(295)은 제 1 주된 캐쉬(294)와 제 1 사전 인출 캐쉬(296)를 유지한다.

컨텐트 서버(20)는 네트워크(25)를 통해 요청을 서빙하는 컴퓨팅 노드이다. 컨텐트 서버(20)는 제 2 프로세서(260), 제 2 메모리(263), 제 2 저장 유닛(265)을 포함한다. 제 2 메모리(263)는, 예를 들어, RAM(random access memory)를 포함할 수 있다. 제 2 저장 유닛(265)은, 이를테면, 직접 액세스 저장 디바이스(direct access storage device : DASD)를 포함할 수 있다. 제 2 메모리(263)는 컨텐트 서버 로직(268) 또는 컨텐트 서버 소프트웨어를 저장한다.

컨텐트 서버(268)의 동작은 도 4를 조회하여 이후에 보다 상세히 설명할 것이다. 컨텐트 서버 로직(268)은 제 2 프로세서(260)에 의한 실행을 위해 제 2 저장 유닛(265)으로부터 제 2 메모리(263)로 로딩되는 컴퓨터 실행가능 코드로서 구현된다. 컨텐트 서버 로직(268)은 제 2 통계 테이블(267), 제 2 통계 테이블을 갱신하는 제 2 갱신 통계 유닛, 사전 인출 힌트 정보를 발생시키는 발생 유닛(266)을 포함한다. 제 2 통계 테이블(267)은 클라이언트에 의해 요청된 정보와 요청된 정보와 연관된 잠재적인 사전 인출 정보에 관한 데이터를 저장한다. 통계 테이블(261)은 클라이언트의 실제 조회 습성에 기초하는 사전 인출 힌트 정보의 발생에 기여한다. 통계 테이블(261)에 관한 세부 사항은 도 38에 도시하고 있다.

제 2 갱신 통계 유닛(269)의 동작은 도 5를 조회하여 보다 상세하게 설명한다. 발생 유닛(266)은 제 2 통계 테이블(267) 내의 정보와 요청된 정보로부터 사전 인출 힌트 정보를 발생시킨다. 발생 유닛(266)은 도 6에 도시한 바와 같은 사전 인출 힌트 정보(prefetch hint information : PHI) 발생 루틴으로서 구현된다.

클라이언트(22)는 제 3 프로세서(240), 제 3 메모리(245), 제 3 저장 유닛(242)을 포함한다. 제 3 메모리(245)는, 예를 들어, RAM을 포함할 수 있다. 제 3 저장 유닛(242)은, 예를 들어, DASD를 포함할 수 있다. 제 3 메모리(245)는 클라이언트 로직(244)이나 클라이언트 소프트웨어를 저장한다. 클라이언트 로직(244)의 동작은 도 7을 조회하여 보다 상세히 설명할 것이다. 클라이언트 로직(244)은 제 3 프로세서(24)에 의한 실행을 위해 제 3 저장 유닛(242)으로부터 제 3 메모리(245)로 로딩되는 컴퓨터 실행가능 코드로서 구현된다.

클라이언트 로직(244)은 클라이언트 요청 처리기(247), 제 2 사전 인출 처리기(241), 제 2 주된 캐쉬(246), 제 2 사전 인출 캐쉬(249)를 포함한다. 클라이언트 객체 요청 처리기(247)는 제 2 주된 캐쉬(246), 제 2 사전 인출 캐쉬(249), 프록시 서버(24), 또는 컨텐트 서버(20)로부터 클라이언트를 위한 요청된 객체를 얻는다. 클라이언트 객체 요청 처리기(247)의 상세한 동작은 도 9에서 도시하고 있다.

클라이언트 로직(244)은 제 2 주된 캐쉬(246)와 제 2 사전 인출 캐쉬(249)를 유지한다. 제 2 주된 캐쉬(246)는 자주 조회되는 객체를 저장하는 데 사용되는 반면, 제 2 사전 인출 캐쉬(249)는 이들이 실제로 조회되기 이전에 객체를 저장하는 데 사용된다. 사전 인출 객체는 조회된 객체와 함께 바로 다음 또는 이전에 조회되지 않은 객체이다. 바람직한 실시예에서, 주된 캐쉬(297)는 통상적인 LRU(least recently used) 알고리즘에 의해 관리된다. 당업자는 임의의 다른 대체적인 방안도 주된 캐쉬(297)를 유지하는 데 적용될 수 있다는 사실을 이해할 것이다.

주된 캐쉬(297)와 사전 인출 캐쉬(298)는 논리적으로 분리될 수 있다. 사전 인출 객체를 주된 캐쉬(297) 내의 실제로 조회된 객체로부터 논리적으로 분리하면, 사전 인출 객체가 너무 많은 저장 공간을 차지하는 것을 방지하게 된다. 이러한 사전 인출 캐쉬와 주된 캐쉬의 논리적인 분리를 달성하는 데에는 다양한 대체적인 방법이 사용될 수 있다. 예를 들어, 단일 캐쉬는 사전 인출 객체의 개수에 부과된 상한(upper limit) 및/또는 사전 인출 객체의 의해 점유된 공간에 부과된 상한을 가질 수 있다. 주된 캐쉬(297)와 사전 인출 캐쉬(298)는 서로 다른 목표를 지원하려 하고 있다. 주된 캐쉬(297)는 반복적으로 조회될 가능성이 있는 인기있는 객체를 캐싱하려고 한다. 사전 인출 캐쉬(298)는 그 세션에서 다음에 조회될 가능성이 가장 큰 객체를 사전 인출함으로써 특정 브라우징 세션을 지원하는 데에만 사용된다. 바람직한 실시예에서, 계층 내의 하위 계층 노드가 상위 계층 노드의 사전 인출 캐쉬 내의 객체에 대한 사전 인출 요청을 발행하면, 그 객체는 하위 계층 노드로 전달되고 사전 인출 목적이 달성되었으므로 그 객체는 상위 계층 노드의 사전 인출 캐쉬(298)로부터 삭제가능하다고 표시될 것이다. 사전 인출 객체가 실제로 조회되면, 사전 인출 객체는 주된 캐쉬(297)의 관리 정책에 따라 주된 캐쉬(297)로 이동할 수 있다. LRU 대체 방안을 이용하는 바람직한 실시예에, 사전 인출 객체는 주된 캐쉬로 이동한다.

35-9

대체적인 표준은 사전 인출 캐쉬(298)로부터 주된 캐쉬(297)로 객체를 수용하도록 고안될 수 있다. 에이브람즈(Abrams) 등의 'Caching Proxies : Limitations and Potentials", Proc. 4th International World Wide Web Conference(1996)에서는, 결정 과정에서 객체 크기를 고려하지 않는 소정의 캐쉬 대체 방안이 논의된 바 있다. 기타 캐쉬 대체 방안은 당업자에게 주지되어 있다.

도 3a는 컨텐트 서버(20) 또는 프록시 서버 계층으로부터 클라이언트를 위해 데이터를 동적으로 사전 인출하는 방법을 나타내는 흐름도이다. 첫 번째로, 블럭(100)에서, 클라이언트는 프록시 서버나 컨텐트 서버(20)로부터 정보를 요청한다. 두 번째로, 블럭(102)에서 컨텐트 서버(20)와 프록시 서버에서의 역사적인 데이터 액세스 또는 데이터 조회 패턴이 식별된다. 실제로, 블럭(100)과 블럭(102)을 완료하는 순서는 바뀔 수도 있다.

세 번째로, 블럭(104)에서, 컨텐트 서버(20)는 블럭(102)에서 식별된 데이터 조회 패턴에 기초하여 요청된 정보로의 관련 액세스에 관한 사전 인출 힌트 정보(PHI)를 발생시킨다. 네 번째로, 블럭(105)에서, 사전 인출 힌트 정보는 요청된 정보에 주석을 달 수 있다. 예를 들어, 사전 인출 힌트 정보는 요청된 정보(즉, 객체)와 함께 운반(piggy-backed)된 메타 데이터(meta data)로서 전달될 수 있다. 사전 인출 힌트 정보를 설정하거나 발생시키는 단계에 대해, 컨텐트 서버(20)는 다수의 공시 사용자 또는 클라이언트로부터의 사용/조회 패턴을 추적한다. 실제로, 컨텐트 서버(20)는 이느 객체 그룹이 요청된 정보 또는 요청된 객체와 관련있는지를 판단한다. 컨텐트 서버(20)는 요청된 정보와 함께 사전 인출 힌트 정보(PHI)를 발생시키고 제공한다. 사전 인출 단계 또는 바로 직전에, 프록시 서버는 지역적 고려 사항에 비추어 PHI를 해석하여 어느 선택 객체를 사전 인출할 지를 결정할 수 있다.

다섯 번째로, 블럭(106)에서, 사전 인출 힌트 정보는 요청된 정보가 프록시 서버 계층을 통과할 때 갱신된다. 블럭(106)에서의 갱신은 수행된 사전 인출 동작, 상위 계층에서의 캐쉬 상태, 지역 조회 패턴과 같은 기타 지역 고려 사항을 반영한다. 프록시 서버는 객체를 다음(하위) 계층 프록시나 클라이언트 스테이션으로 전달하기 이전에 PHI를 수정하여 자신의 사전 인출 결정과 다양한 지역 고려 사항을 반영하도록 한다. 객체가 프록시 계층을 통과할 때, 송신 노드는 초기 PHI를 설정하고, 수신 노드는 초기 PHI로부터 최종 PHI를 유도할 것이다. 최종 PHI는 요청된 객체에 기초하여 어떤 선택 객체를 사전 인출할 지를 정의한다. 사전 인출 동작 및 결정은 수신기에서 개시되며 송신기에서 개시되지 않는다.

마지막으로, 블럭(108)에서는, 사전 인출 힌트 정보와 요청된 정보에 기초하여 선택 정보가 사전 인출된다. 프록시 서버나 클라이언트는 수신된 객체 내의 PHI 정보, (이용가능할 수 있다면) 지역적으로 유지된 조회 정보, 이용가능한 저장 크기, 액세스 시간과 같은 기타 지역적 고려 사항에 기초하여 PHI 내에 명시된 객체의 서브셋(subset)을 사전 인출할지 여부를 결정한다.

블럭(104)에서 사전 인출 힌트 정보를 발생시키는 것은 PICS 프로토콜 또는 PICS 양립 가능 포맷(아래 조회)을 이용하는 것이 바람직하다. 블럭(104)에서 발생한 PHI는 캐쉬된 객체가 요청될 때 객체의 PHI 정보가 조사되어 적절한 사전 인출 동작을 개시할 수 있도록 자신의 메타 데이터의 일부로서 각 캐쉬된 객체와 함께 저장된다. PHI 정보는 PICS 프로토콜을 통해 제공될 수 있다. PICS 프로토콜은 각 PHI 객체의 임계점(criticality)에 관한 정보와 상위 계층에 관한 크기, 캐쉬 상태와 같은 자신의 특성을 제공한다.

사전 인출 힌트 정보를 갱신하는 것은 프록시 서버 계층을 통해 달성될 수 있다. 프록시 서버는 지역 정보를 수집하고, 사전 인출 결정을 프록시 계층으로 분배 또는 분산시킬 수 있다. 또한, 중간 프록시가 객체를 사전 인출한 정보는 저장 및 대역폭 사용을 최적화하기 위해 하위 계층 프록시와 클라이언트 스테이션의 사전 인출 결정에 암시를 줄 수 있다.

프록시 서버 계층 내의 프록시 서버는 수신기 지향 사전 인출 결정(a receiver-oriented prefetch decision)을 제공하기 위해 서로 협력한다. 사전 인출 결정은 각 프록시와 클라이언트에 의해 협력하여 수행될 수 있다. 메타 정보 형식의 사전 인출 힌트 정보는 요청된 정보에 주석을 단다(즉, 운반한다). 사전 인출 힌트 정보는 잠재적인 사전 인출 후보에 관한 정보를 제공한다. 중간 프록시 서버는 지역 조건에 기초하여 메타 정보나 사전 인출 힌트 정보를 수정할 수 있고, 자신이 캐쉬될 때 객체와 함께 메타 정보를 저장하여 객체에 대한 이후의 조회 시 사전 인출이 트리거(trigger)될 수 있도록 한다.

도 3b는 통계 테이블(261) 내의 데이터 포맷을 도시하고 있다. 통계 테이블(261)은 일반적으로 제 1 통계 테이블(293)과 제 2 통계 테이블(267) 중 하나를 조회한다. 하나 이상의 프록시 서버(24)는 제 1 통계 테이블(293)을 유지한다. 컨텐트 서버(20)는 제 2 통계 테이블(267)을 유지할 것이다. 제 1 통계 테이블(293)과 제 2 통계 테이블(267)을 위한 포맷은 실질적으로 유사하거나 동일하여, 프록시 서버 계층 내의 서로 다른 프록시 서버에서의 통계 테이블(261)의 갱신을 촉진하는 것이 바람직하다. 통계 테이블(261)은 통계 테이블(261)과 연관된 프록시 서버나 컨텐트 서버(20)의 관점에서 실제 관찰자 조회 습성(viewer reference behavior)에 기초하여 사전 인출 힌트 정보(PHI)를 제공하는 것을 돕는다. 통계 테이블(261)은 프록시 서버 계층 내에서 자신보다 하위인 통신 클라이언트 모두의 요청을 관찰할 수 있다.

예컨대, 도 1에서, 제 1 계층-3 프록시 서버(50) 내의 제 1 통계 테이블(293)은 제 1 클라이언트(600)와 제 2 클라이언트(603)의 통합 데이터 액세스 습성을 추적한다. 한편, 제 2 계층-3 프록시 서버(55) 내의 제 1 통계 테이블(293)은 제 3 클라이언트(700)와 제 4 클라이언트(703)의 통합 데이터 액세스 습성을 추적한다. 제 1 계층-2 프록시 서버(40) 내의 제 1 통계 테이블은 제 1 클라이언트(600), 제 2 클라이언트(603), 제 3 클라이언트(700), 제 4 클라이언트(703)의 통합 습성을 추적한다. 제 1 계층-3 프록시 서버(50)와 같은 하위 계층 프록시가 클라이언트(600)의 관심을 더 잘 추적할 수도 있지만, 하위 계층 프록시는 충분한 샘플 크기의 클라이언트로부터 충분한 조회 통계를 얻어서 의미있는 사전 인출 힌트 정보(PHI)를 제공할 수 없을 수도 있다.

바람직한 실시예에서, 객체 0가 컨텐트 서버(20)로부터 요청되면, 서버(20)는 자신의 제 2 통계 테이블(267) 내의 정보에 기초하여 초기 사전 인출 힌트 정보 PHI를 제공한다. 객체 0가 프록시 계층을 따라 아래로 전달되면서, 클라이언트(22)로 가는 경로 상에 있는 임의의 프록시 서버(24)는 객체 0에 대한 충분한 지역 조회가 있는 경우에는 객체 헤더 내의 사전 인출 힌트 정보(PHI)를 수정하여 제 1 통계 테이블(293) 내의 지역 상황에 PHI를 맞출 수 있다. 제 1 통계 테이블(293)은 하위 계층 프록시 서버에 의한 특정 클

35-10

라이언트의 선호도 또는 잘 색인된 관심(well-indexed interests)과 같은 인자에 기초하여 제 2 통계 테이블(267)을 대체(supersede)할 수 있다.

바람직한 실시예에서, 컨텐트 서버(20)는 제 2 통계 테이블(267)을 유지하는데, 이는 요청된 정보에 대한 클라이언트(22)의 데이터 액세스 빈도의 데이터베이스이다. 데이터베이스는 요청하는 프록시 서버(24)의 데이터 액세스 빈도를 포함하도록 확장될 수 있다. 마찬가지로, 하나 이상의 프록시 서버(24)가 제 1 통계 테이블(293)을 유지한다. 이를테면, 통계 테이블(261)은 제 1 데이터 구조(304)와 제 2 데이터 구조(306)를 갖는 제 1 어레이(383)를 구비한다. 도시한 바와 같이, 제 1 데이터 구조(304)와 제 2 데이터 구조(306)는 기록된다. 통계 테이블(261) 내의 각 기록은 부모 객체 식별자(301)(즉, ObjectID), 제 1 카운트(302)(즉, TCount), 액세스 리스트 데이터 포인터(303)(즉, Alist)를 포함하는 수 개의 필드로 이루어진다.

부모 객체 식별자 필드(301)는 객체 식별자를 포함한다. 예를 들어, 부모 객체 식별자(301)는 인터넷이나 관련 애플리케이션의 URL을 포함한다. 제 1 카운트 필드(302)는 서버에 의해 조회된 부모 객체 식별자와 연관된 부모 객체의 회수를 포함한다. 즉, 제 1 카운트 필드(302)는 요청된 객체의 실제 요청 빈도를 포함한다. 제 1 카운트(302)는 부모 객체 액세스 빈도를 나타낸다. 액세스 리스트 데이터 포인터(303)는 액세스 리스트(370)에 대한 데이터 포인터를 포함한다.

통계 테이블(261)은 액세스 리스트(370)를 포함하는데, 이는, 예를 들어, 제 1 액세스 기록(340), 제 2 액세스 기록(350), 제 3 액세스 기록(360), 제 4 액세스 기록(380)을 포함한다. 도시한 바와 같이, 제 1 데이터 구조와 연관된 부모 객체 식별자는 제 1 액세스 기록(340), 제 2 액세스 기록(350), 제 3 액세스 기록(360)의 링크 리스트(linked list)로 이루어진 액세스 리스트(370)를 포함한다. 다음 기록 데이터 포인터 필드(313)는 특정 부모 객체 식별자(301)와 연관된 액세스 리스트(370) 상의 다음 액세스 기록에 대한 포인터를 포함한다. 제 2 데이터 구조(306)와 연관된 부모 객체 식별자(301)에 대한 액세스 리스트는 제 4 액세스 기록(308)만을 포함한다.

액세스 리스트 상의 각 기록 또는 데이터 구조는 자녀 객체 식별자(310)(즉, AObjectID), 제 2 카운트(312)(즉, ACount), 다음 기록 데이터 포인터(313)(즉, Nxt)를 포함하는 수 개의 필드로 이루어져 있다. 데이터 구조는 일반적인 데이터 구조, 기록, 어레이 형태일 수 있다. 도 4에 도시한 바와 같이, 데이터 구조는 액세스 리스트(370) 내에 링크 리스트를 형성하는 기록이다.

대체적인 실시예에서, 통계 테이블의 데이터 포맷은 트리 구조 내에 있는 다수의 교대 2차 어레이(alternate secondary arrays)와 관련된 1차 어레이(primary array)를 포함할 수 있다. 1차 어레이는 부모 객체와 관련된 정보를 포함할 것이고, 2차 어레이는 대응 부모 객체의 자녀 객체에 관한 정보를 포함할 것이다.

자녀 객체 식별자 필드(310)는 객체의 객체 식별자(URL)를 포함한다. 부모 객체 식별자(301)는 액세스 리스트 상의 하나 이상의 자녀 객체 식별자(310)와 관련된다. 자녀 객체 식별자(310)는 자녀의 부모 객체가 실재로 조회되는 경우 잠재적인 사전 인출 후보를 식별한다. 제 2 카운트(312)는 자신의 부모 객체를 통해 자녀 객체가 조회된 회수를 포함한다. 달리 말해서, 제 2 카운트는 자녀 객체가 연관된 부모 객체의 이전 조회 바로 다음에 후속적, 일시적으로 조회된 회수이다. 제 2 카운트(312)는 자녀 객체 액세스 빈도를 나타낸다.

액세스 리스트(370) 상의 자녀 객체 식별자(310)는 클라이언트의 실제 조회 습성을 관찰하여 동적으로 얻을 수 있다. 예를 들어, HTML에서, 웹 페이지는 다른 웹 페이지로의 HTTP 링크를 구비할 수 있다. HTTP 링크에 의해 표시된 이들 웹 페이지는 HTTP 링크를 포함하는 (부모) 웹 페이지에 대한 잠재적인 자녀 객체이다. 부모 객체가 관찰될 때, 부모 객체 내에 포함된 조회(HTTP) 링크가 클릭(click)되면, 대응 자녀 객체가 조회된다. 자녀 객체에 대한 조회 링크가 다수의 다른 부모 객체 내에 포함될 수 있으므로, 자녀 객체는 하나 이상의 부모를 가질 수 있다. HTTP 프로토콜 하에서, 자녀 객체에 대한 요청은 자신의 헤더 내의 조회자 정보(referrer information)로서 부모 URL을 포함한다. 따라서, 서버는 (자녀 헤더 내의 조회자 정보를 통해) 후속적인 자녀 조회를 자신의 부모에 동적으로 관련시킬 수 있다. 클라이언트가 자녀 객체를 자주 클릭하거나 선택하면, 자녀 객체는 사전 인출 후보나 선택 정보가 될 것이다.

관찰자 조회 습성을 동적으로 추적할 뿐 아니라, 웹 로그(Web log) 상의 정적 분석 또는 발굴(mining)도 주기적으로 수행되어 통계 테이블(261)을 위한 객체 그룹에 관한 정보를 유도할 수 있다. 발굴 알고리즘은 함께 조회된 객체 그룹을 식별하여 PHI를 위한 잠재적인 사전 인출 후보를 얻는다. 웹 로그를 발굴하여 조회 패턴을 유도하는 한 방법은 건룽 우(Kun-lung Wu) 등에 의한 "SpeedTracer : A Web Usage Mining and Analysis Tool", IBM Research Report 20895, (1997년 5월)(IBM Systems Journal v37, n1, p89-105, 1998 조회)에 설명되어 있다. SpeedTracer 알고리즘은 사용자 세션에서 함께 조회된 상습 페이지 그룹과 상습 통과 경로에 관한 정보를 제공한다. SpeedTracer 알고리즘은 즉시 함께 조회되지 않는 관련 조회를 드러낼 수 있는 면밀한(in-depth) 유형의 발굴이다. 또한, 주어진 객체로의 통과 경로에 기초하여 조회된 객체의 다음 세트에 관해 보다 나은 예측을 할 수도 있다.

다른 유형의 데이터 발굴 알고리즘은 통계 테이블(261)에 대한 데이터 액세스 패턴을 식별하는 데 사용될 수 있다. 예컨대, 다른 데이터 발굴 알고리즘은 조회 패턴을 얻기 위한 웹 로그 분석에도 응용될 수 있는 분류 및 군집(classification and clustering)을 포함한다. 또한, PHI는 자신의 객체 관계에 대한 지식에 기초하여 컨텐트 제공자에 의해 공급될 수도 있다. 이를테면, PHI는 요청된 소프트웨어 다운로드, 요청된 JAVA 애플릿을 위한 자바 클래스, 요청된 객체 내에 포함된 (GIF와 같은) 객체를 인출하고 업그레이드(upgrade)할 수 있지만, 여기에 국한되지는 않는다.

바람직한 실시예에서, 통계 자료를 수집할 때, 클라이언트 스테이션에 의한 객체의 실제 조회는 객체에 대한 사전 인출 요청과는 구별된다. 통계 테이블(261) 내의 제 1 카운트(302)(즉, TCount)와 제 2 카운트(310)(즉, RCount)는 사전 인출이 아닌 실제 조회를 반영한다. 즉, 사전 인출 동작으로 인해 제 1 통계 테이블(293)이나 제 2 통계 테이블(267)을 갱신하게 되지 않으며, 실제 객체 조회만이 이들을 갱신할 것이다.

35-11

사전 인출 힌트 정보나 사전 인출 힌트 정보 리스트는 요청된 정보나 요청된 객체에 주석을 달기 위해 다양한 데이터 구조를 사용할 수 있다. 예를 들어, 사전 인출 힌트 정보는 운반된 메타 데이터를 이용하여 요청자와 서버 노드 사이에서 통신할 수 있다. HTTP 구현에서, 정보 교환은 기존의 웹 프로토콜을 이용하여 객체 헤더 내에 있는 메타 데이터로서 포함될 수 있다. PICS(Platform for Internet Content Selection)는 전자 컨텐트(electronic content)에 관한 메타 정보를 전송하는 방법을 명시한다. PICS는 (1) 요청된 정보에 주석 달기, (2) 선택 정보, 사전 인출 후보, 또는 자녀 객체의 사전 인출, (3) 정보를 사전 인출한 후 사전 인출 동작 승인에 적용할 수 있다.

PICS는 당업자에게는 웹 컨소시엄 프로토콜 권장(Web Consortium Protocol Recommendation)으로 알려져 있다. PICS는 "How much nudity is associated with this content"와 같은 가치 참조 평가 라벨(values-based rating labels)을 송신하기 위해 먼저 사용되었지만, 메타 정보의 포맷과 의미는 전적으로 일반적이다. PICS에서 전자 컨텐트에 관한 메타 정보는 "평가 서비스(rating services)"나 정보의 생산자-및-의도된-사용(producer-and-intended-usage)에 따라 그룹화되고, 이러한 하나의 그룹 내에서, 임의 개수의 카테고리나 치수의 정보가 전송될 수 있다. 각 카테고리는 허용된 값의 범위를 갖고, 특정 컨텐트에 대해 특정 카테고리는 단일한 값 또는 다수의 값을 가질 수 있다. 또한, ("PICS 라벨"로 알려진) 메타 정보 그룹은 만기 정보(expiration information)를 포함할 수 있다. 또한, PICS 라벨이 하나 이상의 전자 컨텐트에 적용되도록 하는 설비도 있다. 특정 전자 컨텐트를 위한 각각의 PICS 라벨은 독립적으로 컨텐트에 추가되거나 컨텐트로부터 제거될 수 있다.

예를 들어, 영상 파일은 자신의 "평가 서비스" 필드가 자신이 "SafeSurf" 평가 시스템에 따른 가치 참조 평가 라벨을 포함하고 있음을 나타내는 PICS 라벨과 함께 서버로부터 전달될 수 있다. 본 발명의 바람직한 실시예에 따르면, 파일이 기획 프록시(enterprise proxy)를 통과할 때, 영상 파일은 PICS 라벨을 위한 새롭게 갱신된 카테고리 값을 갖게 되어, "평가 서비스"의 지역적 고려 사항을 반영할 수 있다. 따라서, 클라이언트 컴퓨터는 PICS 라벨의 새로운 카테고리 값만을 보게 된다. HTTP프로토콜은 자신의 요청 헤더와 응답 헤더가 PICS를 지원하도록 증가(augment)되었다. NNTP와 같은 기타 일반적인 애플리케이션 프로토콜을 규정하는 기술적인 개체(technical bodies)도 이제 PICS 지원을 추가할 것을 고려하고 있다. 이들 프로토콜의 일부로서, PICS 라벨 유형 리스트가 요청과 함께 포함될 수 있다. 또한, PICS는 중앙 라벨 사무국 서버(central label bureau server)로부터 PICS 정보를 요청하는 문의 포맷을 명시하고 있다.

샘플 PICS 라벨은 다음과 같으며,

("http://the.content" exp "1997.07.09T08:15-0500" r (n 4 s 3 v 2)을 위한 PICS-1.1 "http://the.rating.service" 라벨)

여기서, "n", "s", "v"는 다양한 메타 정보 유형의 전송 이름이고, 이 컨텐트에 대한 응용가능 값은 (n에 대해) 4, (s에 대해) 3, (v에 대해) 2이다. ID "http://the.rating.service"를 인식하는 소프트웨어만이 이들 카테고리와 값을 해석하는 방법을 알 것이다.

바람직한 실시예에서는, 3 개의 다른 종류의 PICS가 도입된다. 첫 번째 종류의 PICS 라벨은 사전 인출 라벨(즉, P 라벨)이라 지칭하며, PHI를 제공하기 위해 서버 노드에 의해 사용된다. 두 번째 종류의 PICS 라벨은 요청 라벨(즉, R 라벨)이라 지칭하며, 사전 인출 후보 객체를 요청하는 데 사용된다. 세 번째 종류의 PICS 라벨은 조회 라벨(X 라벨)이라 지칭하며 어느 사전 인출 객체가 실제로 조회되었는가에 기초하여 사전 인출의 효율에 관한 피드백을 제공한다.

사전 인출 라벨은 식별자 카테고리, 확률 카테고리, 크기 카테고리, 캐싱 카테고리를 포함하는 다음 카테고리의 임의의 조합을 포함할 수 있으나, 여기에 국한되지는 않는다. 식별자 카테고리(즉, ID 카테고리)는 사전 인출 후보의 URL을 전달하는 값을 갖는다. 확률 카테고리(즉, PROB 카테고리)는 사전 인출의 유용성의 추정치를 전달하는 값을 갖는다. 바람직한 실시예에서는 조회 확률의 추정치가 사용된다. 조회 확률은 사전 인출 후보가 조회되는 확률을 나타낸다 조회 확률은 사전 인출 후보가 조회될 확률을 나타낸다. 크기 카테고리는 사전 인출 후보의 크기 또는 메모리 조건을 전달하는 값을 갖는다. 캐싱 카테고리는 자신의 제 1 주된 캐쉬(294)나 자신의 제 1 사전 인출 캐쉬(296) 내에 사전 인출 후보 객체의 복사본을 갖는 임의의 상위 계층 프록시 서버에 관한 정보를 전달하는 값을 갖는다. 바람직한 실시예에서는 가장 가까운 상위 계층 프록시만이 식별된다.

요청 라벨은 식별자 카테고리(즉, ID 카테고리)를 포함하는 다음 카테고리를 포함할 수 있으나, 여기에 국한되지는 않는다. 식별자 카테고리 값은 사전 인출될 객체의 URL을 전달한다. 바람직한 실시예에서, (프록시 또는 클라이언트) 노드가 제시된 사전 인출 후보 객체를 사전 인출하려고 하면, HTTP 헤드 요청이 요청 라벨을 이용하여 프록시 계층을 통해 컨텐트 서버(20)로 돌려보내져서 사전 인출될 객체의 URL을 전달한다. 프록시 서버(24)가 하위 계층 노드로부터 사전 인출 요청을 수신할 때, 객체가 자신의 주된 캐쉬나 사전 인출 캐쉬 내에 있으면 그 요청을 서빙할 것이다. 그렇지 않으면, 사전 인출 요청을 다음 상위 계층 프록시로 전달한다.

조회 라벨은 객체 식별자 카테고리(Object-ID 카테고리)와 조회 식별자 카테고리(Referrer-ID 카테고리)를 포함하는 다음 카테고리를 포함할 수 있지만, 여기에 국한되지는 않는다. 객체 식별자 카테고리는 조회된 사전 인출 객체의 URL을 전달하는 값을 갖는다. 조회 식별자 카테고리는 객체 식별자 카테고리 값이 나타내는 자신의 PHI 객체 내에 포함된 객체의 URL을 전달하는 값을 갖는다.

바람직한 실시예에서, 사전 인출 객체가 실제로 조회되면, HTTP 헤드 요청이 조회 라벨을 이용하여 프록시 계층을 통해 컨텐트 서버(20)로 돌려보내져서, 조회된 사전 인출 객체와 자신의 조회자 객체(referrer object)의 URL을 전달한다. 이러한 유형의 HTTP 헤드 요청을 이후로부터는 사전 인출 조회 메시지로 지칭한다. 프록시 서버(24)가 하위 계층 노드로부터 사전 인출 조회 메시지를 수신하면, 자신의 제 1 통계 테이블(293)을 갱신하고 그 요청을 다음 상위 계층 프록시 서버로 전달한다.

도 4는 컨텐트 서버(20) 내의 제 2 통계 테이블(267)을 갱신하는 서버 로직(260)의 예를 도시하고 있다. 제 2 통계 테이블(267)의 갱신은 클라이언트(22)로부터의 객체 요청에 의해 트리거된다. 컨텐트 서버(20)는 요청된 객체를 클라이언트(22)에게 보내기 이전에 객체의 객체 헤더로의 주석 달기 또는 삽입을 위한

사전 인출 힌트 정보를 발생시키는 것이 바람직하다.

조회 블럭(405)에서 시작하여, 컨텐트 서버(20)는 입력을 기다린다. 판단 부호(410)에서 수신된 입력이 객체 O에 대한 객체 요청이면, 통계 갱신 유닛(252) 또는 제 2 갱신 통계 루틴(269)이 조회 블럭(415) 내에서 호출된다. 조회 블럭(420)에서는, 사전 인출 힌트 정보(266)를 발생시키는 발생 유닛(266) 또는 사전 인출 발생 루틴이 호출된다. 발생 유닛(266)이나 PHI 발생 루틴의 상세한 설명은 도 6을 조회하여 이후에 설명하겠다. 조회 블럭(425)에서, PHI는, 예를 들어, 요청된 객체의 객체 헤더로 삽입된다. 그렇지 않으면, 사전 인출 힌트 정보는 블럭(425)에서 요청된 객체 또는 요청된 정보에 주석을 단다. 바람직한 실시예에서는 PICS 프로토콜을 사용하여 요청된 객체에 주석을 단다. PHI는 P 라벨 내에서 포착(capture)되는데, 여기서 식별자 카테고리 값은 사전 인출 후보의 URL을 반영한다. 블럭(430)에서, 요청된 객체는 요청하는 클라이언트나 요청하는 프록시 서버로 돌아간다. 판단 부호(435)에서, 수신된 입력이 사전 인출 조회 메시지(즉, 사전 인출 객체 O가 실제로 조회되었음을 나타내는 X 리벨을 갖는 HTTP 헤드 요청)이면, 통계 갱신 수단(252) 또는 제 2 갱신 통계 루틴(269)이 단계(440)에서 호출된다. FTP(file transfer protocol) 요청과 같은 다른 형태의 입력에 대한 것은 본 발명의 초점이 아니며, 조회 블럭(445)에서 적절한 기타 처리기가 호출될 수 있다.

도 5는 통계 갱신 유닛(252)이 상기 통계 테이블(261) 중 하나를 갱신하는 방법을 도시하고 있다. 통계 갱신 유닛(252)은 제 1 통계 테이블(293)을 갱신하는 제 1 갱신 통계 루틴(264)이나 제 2 통계 테이블(267)을 갱신하는 제 2 갱신 통계 루틴(269), 또는 제 1 갱신 통계 루틴(264)과 제 2 갱신 통계 루틴(269) 모두를 조회한다. 통계 테이블(261)은 제 1 통계 테이블(293)이나 제 2 통계 테이블(267), 또는 제 1 통계 테이블(293)과 제 2 통계 테이블(267) 모두를 조회한다. 통계 갱신 유닛(252)은 먼저 조회자 객체 내의 정보를 찾는다. 이어서, 요청된 객체는 액세스 리스트나 조회자 객체의 사전 인출 리스트 상에 배치된다.

조회 블럭(505)에서 시작하여, 통계 테이블(261)은 제 1 카운트를 증가시킴으로써 객체 O가 한번 더 조회되었음을 반영하도록 갱신된다. 달리 말해서, 블럭(505)의 TCount(O)가 1만큼 증가한다. 객체 O가 통계 테이블(261) 내에 아직 존재하지 않으면, TCount(O)를 1로 초기화하여 부모 객체 식별자가 통계 테이블로 삽입된다. 통계 테이블(261) 내에 이용가능한 빈 슬롯이 없으면, 새로운 부모 객체 식별자는 가장 먼저 조회된 엔트리를 대체한다. 판단 부호(510)에서, 요청된 객체는 예를 들어, 요청된 객체의 객체 헤더를 조사함으로써 조회자 정보를 위해 조사된다. 이를테면, HTTP 프로토콜에서, 조회자 객체가 헤더 내에 제공되어, 어느 객체가 요청된 객체로의 HTTP 링크를 포함하는지를 나타낸다. 조회자 객체(R)가 헤더 내에서 발견되면, 단계(520)에서 객체(O)가 이미 객체(R)의 액세스 리스트 상에 존재하는지는 알기 위해 통계 테이블(261)이 조사된다. 객체(O)가 이미 존재하면, 블럭(530)에서 요청된 객체에 대한 제 2 카운트(즉, RCount(O))가 1만큼 증가한다. 그렇지 않으면, 제 2 카운트(즉, RCount(O))를 1로 초기화하여 블럭(525)에서 요청된 객체(즉, 객체 O)가 부모 객체(즉, 객체 R)의 액세스 리스트로 삽입된다.

도 6은 사전 인출 힌트 정보를 사전 인출 힌트 정보 리스트로 발생시키는 발생 유닛(266)의 예를 도시하고 있다. 사전 인출 힌트 리스트는 제 1 카운트에 대한 제 2 카운트의 비율이 힌트 참조값을 초과하면 자녀 객체의 자녀 객체 식별자를 포함한다. 사전 인출 힌트 정보 리스트는 정제된, 또는 필터링된 액세스 리스트를 나타낸다. 즉, 발생 유닛(266)은 통계 테이블(261)로부터 입력을 수용하고 입력을 필터링하여 사전 인출 힌트 정보 리스트를 얻는다.

사전 인출 힌트 정보를 발생시키는 발생 유닛(266)은 조회 블럭(610)에서 시작되는 사전 인출 힌트 정보(PHI) 발생 루틴(266)을 나타낸다. 조회 블럭(610)에서, 부모 객체(즉, 객체 O)의 액세스 리스트가 확인되어 리스트가 비었는지(즉, Alist(O)가 0인지)를 알기 위해 확인된다. 액세스 리스트가 비어있지 않으면, 조회 블럭(620)에서 Cj를 부모 객체(즉, 객체 O)의 액세스 리스트 상에서 식별된 다음 자녀 객체(즉, 자녀 객체 식별자)로 설정한다.

판단 부호(630)에서, 자녀 객체 Cj의 조회 확률은 사전결정된 참조값(예컨대, 0.75)인 힌트 참조값(즉, TH)과 비교된다. 바람직한 실시예에서, Cj의 조회 확률은 제 1 카운트에 대한 제 2 카운트의 비율(즉, RCount(Cj)/TCount(O))로 정의된다. 이 비율은 부모 객체(객체 O)가 조회된 후 자녀 객체(객체 Cj)가 조회될 확률의 추정치이다. 판단 부호(630)에서, Cj의 조회 확률이 힌트 참조값(즉, TH)보다 크면, 자녀 객체 식별자(Cj)는 조회 블럭(640)에서 PHI 리스트 내에 포함될 것이다. 달리 말해서, 비율이 힌트 참조값을 초과하면, 자녀 객체 식별자와 연관된 정보가 사전 인출 힌트 정보 리스트 상에 배치된다.

발생 유닛(266)은 조회 블럭(630)에서의 비교를 위해 소프트웨이 서브루틴이나 소프트웨어 구성요소를 요청할 수 있다. 특히, 사전 인출 힌트 정보를 사전 인출 힌트 정보 리스트로 발생시키는 발생 유닛(266)은 조회 확률을 계산하는 계산 수단과 조회 확률을 힌트 참조값과 비교하는 비교 수단을 포함할 수 있다.

바람직한 실시예에서, PICS 프로토콜을 사용하여 PHI 정보가 P 라벨 내에 지장되는데, 자녀 객체(Cj)의 URL은 식별자 카테고리(즉, ID 카테고리 값)로서 취해지고, 그 참조 확률과 크기는 확률 카테고리(즉, PROB)와 크기 카테고리 값으로서 각각 취해진다. 판단 부호(645)에서, 객체 O의 액세스 리스트가 완전히 스캐닝되지 않으면, 조회 블럭(620)이 재실행될 것이다.

사전 인출 후보를 통계 테이블(261)로부터 선택하기 위해 다양한 대체적인 참조가 사용될 수 있다. 예를 들어, 과거 시간 간격을 통한 조회 확률의 변화하는 평균이 참조으로서 사용될 수도 있다. 또한, 사전 인출 후보의 선택 과정을 반복적으로 만들 수도 있다. 자녀 객체(객체 Cj)가 부모 객체(객체 O)의 사전 인출 후보로 선택되면, 자녀 객체(객체 Cj)의 액세스 리스트 상에 있는 임의의 자녀 객체(객체 Cji)는 부모 객체(객체 O)의 사전 인출 리스트 내에 포함되지 않았다면 포함될 지 여부를 판단하기 위해 평가될 수 있다. 판단 부호(630)에서, 비교 유닛이나 평가 유닛은 이제 RCount(Cj)/TCount(O)와 RCount(Cji)/TCount(Cj)의 결과가 힌트 참조값(TH)보다 큰지를 테스트한다.

도 7은 클라이언트 로직(244)의 동작의 예를 도시하고 있다. 도 7은 객체가 요청되면 클라이언트 객체 요청 처리기(247)가 호출되는 것을 도시한다. 그렇지 않으면, 사전 인출 처리기(241)가 요청되지 않은 객체를 처리하는데, 클라이언트(22)의 제 2 사전 인출 캐쉬(249) 내에 저장될 필요는 없다.

블럭(710)에서 시작하여, 클라이언트(22)는 입력을 대기한다. 판단 부호(720)에서, 수신된 입력이 객체이면, 제 2 사전 인출 처리기(241)가 조회 블럭(730)에서 호출된다. 블럭(740)에서는, 클라이언트 객체 처리기가 호출되어 객체를 클라이언트(22)에서 요청하는 애플리케이션으로 반환한다. 판단 부호(750)에서 수신된 입력이 객체 요청이면, 조회 블럭(760)에서 클라이언트 객체 요청 처리기(247)가 호출된다. (푸쉬(push) 요청과 같은) 다른 유형의 입력에 대해서는, 본 발명의 논점이 아니며, 적절한 기타 처리기가 조회 블럭(770)에서 호출될 수 있다.

도 8은 제 1 사전 인출 처리기(248)나 제 2 사전 인출 처리기(241), 또는 제 1 사전 인출기(248)와 제 2 사전 인출 처리기(241) 모두를 포함하는 사전 인출 유닛(250)의 동작의 예를 도시하고 있다. 간단히 말해서, 사전 인출 유닛(250)은 도 6의 발생 유닛(266)에 의해 발생한 사전 인출 리스트를, 사전 인출 힌트 정보 리스트 상에서 조회된 자녀 객체가 사전 인출 캐쉬(298) 중 하나로의 저장을 보장하는지를 판단하는 입력으로 사용한다. 사전 인출 유닛(250)은 사전 인출 힌트 정보 리스트 상의 각 사전 인출 객체 후보나 자녀 객체 식별자에 대한 사전 인출 값을 계산한다. 사전 인출 값은 자녀 객체의 액세스 확률과 자녀 객체를 저장하는 액세스 비용을 고려한다. 최고 참조값을 갖는 자녀 객체는 사전 인출 캐쉬(298) 중 하나 내의 이용가능한 공간이 찰 때까지 먼저 캐싱된다.

조회 블럭(805)에서 시작하여, L을 사전 인출 후보 객체의 PHI 리스트로 설정한다. 사전 인출 후보 객체는 사전 인출 힌트 정보 리스트 상의 자녀 객체 식별자의 관점에서 표현될 수 있다. 지역 캐쉬(즉, 사전 인출 캐쉬 또는 주된 캐쉬) 내에 나타난 임의의 사전 인출 후보는 L로부터 제거된다. 블럭(810)에서는, L 내의 각 객체의 사전 인출 값(PV)이 계산된다. 바람직한 실시예에서, PV값은 조회 확률과 액세스 비용의 곱으로 정의된다. 액세스 비용은 PHI 리스트 상에서 식별된 사전 인출 후보 객체나 자녀 객체를 캐싱하는 가장 가까운 사이트(site)에 기초하여 추정된다. 실제적인 문제로서, 자녀 객체는 전체 자녀 객체로서보다는 자녀 객체 식별자에 의해 PHI 리스트 상에서 식별되어 저장 공간을 절약하는 것이 바람직하다. 조회 확률은 확률 카테고리(PROB 카테고리) 값으로부터 구하고, 가장 가까운 캐싱 사이트는 PICS 라벨의 캐싱 카테고리 값으로부터 구하는 것이 최적이다. 블럭(820)에서는, 최고 PV 값을 갖는 사전 인출 후보 객체가 선택된다. 블럭(830)에서는, 사전 인출 객체 후보(즉, Oj)를 수용하는 데 이용가능한 사전 인출 버퍼 공간의 양이 판단된다. 이 공간은 평가되고 있는 현재 사전 인출 객체 후보(즉, Oj)의 PV 값의 일부분(예컨대, 절반)인 보다 낮은 PV 값을 갖는 객체에 의해 현재 사용되거나 점유되고 있지 않다. 사전 인출 캐쉬(298) 중 하나에서 삭제가능한 것으로 표시된 객체에 의해 점유된 임의의 저장 공간은 현재 사용 중이 아닌 것으로 간주된다. 판단 부호(840)에서, 사전 인출 캐쉬(298) 중 하나를 조사하여 Oj를 위한 충분한 공간이 존재하는지를 확인한다. 충분한 공간이 존재하면, 조회 블럭(850)에서 객체 Oj가 사전 인출 캐쉬(298) 중 하나로 사전 인출된다. 블럭(860)에서는 Oj가 PHI 리스트(즉, L)로부터 제거된다. 조회 블럭(870)에서, L이 비어있지 않으면, 조회 블럭(820)과 연관된 단계가 재실행된다.

바람직한 실시예에서는, 각 객체가 한 번에 하나씩 사전 인출되지만, 하나의 요청에 따라 다수의 객체가 함께 사전 인출될 수도 있다. 조회 블럭(850)에서는, Oj가 사전 인출 리스트 상에 오를 수 있다. 블럭(870)에서, L이 비었으면, 사전 인출 요청이 전체 사전 인출 리스트와 함께 발생한다. 또한, 사전 인출 캐쉬 내에 충분한 저장 공간이 존재하지 않아도, 대기 사전 인출 리스트(a pending prefetch list)가 유지되어, 저장이 가능할 때 추가 사전 인출을 개시할 수도 있다.

HTTP 프로토콜을 이용함으로써, 사전 인출 중에 통신 세션을 개방한 채로 유지하여 통신 오버헤드를 줄일 수 있다. 각 사전 인출에 대해 세션을 재설정할 필요는 없다.

도 9는 클라이언트 객체 요청 처리기(247)의 동작의 예를 도시하고 있다. 요약하면, 클라이언트 객체 요청 처리기(247)는 먼저 제 2 주된 캐쉬(246)를 검색하고, 다음에 제 2 사전 인출 캐쉬(249)를 검색한다. 객체가 제 2 주된 캐쉬(246) 내에도 존재하지 않고, 제 2 사전 인출 캐쉬(249) 내에도 존재하지 않으면, 객체 요청이 서버로 전달된다. 객체가 제 2 사전 인출 캐쉬(249) 내에 존재하면, 사전 인출 조회 메시지가 서버 중 하나로 전달되어 사전 인출 객체가 실제로 조회되었음을 나타낸다. 그런 다음, 제 2 사전 인출 처리기(241)와 클라이언트 객체 처리기가 사용된다.

판단 부호(905)에서 시작하여, 요청된 객체 O가 제 2 주된 캐쉬(246) 내에서 발견되면, 제 2 사전 인출 처리기(241)가 조회 블럭(925)에서 호출되어 객체 O와 연관된 PHI에 기초하여 적절한 사전 인출 동작을 개시한다. 조회 블럭(930)에서는, 클라이언트 객체 처리기가 호출되어 클라이언트 측의 요청 애플리케이션으로 요청된 객체를 반환한다. 판단 부호(910)에서, 요청된 객체 O가 제 2 사전 인출 캐쉬(249)에서 발견되면, 자신의 제 2 통계 테이블(267)을 갱신하여 사전 인출 객체의 실제 조회를 포착할 수 있도록, 블럭(920)에서 사전 인출 조회 메시지가 컨텐트 서버(20)로 전달된다. 사실 상, 사전 인출 조회 메시지가 프록시 계층을 통해 컨텐트 서버(20)로 전달될 때, 각각의 중간 프록시 내의 제 2 통계 테이블(267)도 갱신될 것이다. 블럭(910)에서, 객체가 제 2 사전 인출 캐쉬(249) 내에 존재하지 않으면, 조회 블럭(915)에서 객체 요청이 프록시 계층을 통해 컨텐트 서버(20)로 전달된다.

사전 인출 조회 메시지는 일괄 처리(batch)될 수 있다. 즉, 사전 인출 조회 리스트가 유지될 수 있고, 주기적으로 합성 사전 인출 조회 메시지가 상위 계층 서버로 전달되어, 사전 인출 조회 리스트 상의 모든 객체의 조회를 승인한다.

도 10은 프록시 서버 로직(295)의 동작의 예를 도시하고 있다. 간단히 말해서, 입력이 객체 요청인지, 객체 전송인지, 사전 인출 조회 메시지인지에 따라 입력이 저장된다. 입력에 따라, 3 가지 서로 다른 처리기 또는 소프트웨어 루틴, 즉, 프록시 객체 요청 처리기(283), 프록시 객체 처리기(284), 사전 인출 조회 메시지 처리기(285)를 이용할 수 있다.

조회 블럭(1005)에서 시작하여, 프록시 서버(24)는 입력을 대기한다. 판단 부호(1010)에서, 수신된 입력이 객체 O에 대한 객체 요청이면, 프록시 서버(24)는 블럭(1015)에서, 그 객체 O에 관해 대기 중인 요청이 이미 존재하는지를 확인한다. 대기 중인 요청은 이 프록시 서버(24)에 의해 발생한 그 객체 O에 대한 사전 인출 요청 또는, 프록시 서버 계층에서 하위인 다른 프록시 서버 노드로부터 기인한 객체 O에 대한 이전 요청으로 인한 것일 수 있다. 조회 블럭(1015)에서 객체 O에 대해 대기 중인 요청이 존재하지 않으면, 프록시 객체 요청 처리기(283)가 블럭(1020)에서 호출된다. 판단 부호(1025)에서, 수신된 입력이 (상위

35-14

계층 프록시로부터의) 객체이면, 프록시 객체 처리기(284)가 조회 블럭(1030)에서 호출된다. 그렇지 않으면, 블럭(1040)에서, 수신된 블럭이 사전 인출 조회 메시지이면, 사전 인출 조회 메시지 처리기(285)가 블럭(1045)에서 호출된다. (FTP 요청과 같은) 다른 유형의 입력에 대해서는, 본 발명의 논점이 아니며, 블럭(1060)에서 적절한 기타 처리기가 호출될 수 있다.

도 11은 프록시 객체 요청 처리기(283)의 동작의 예를 도시하고 있다. 요약해서, 객체가 사전 인출 캐쉬 내에 있으면, 그 객체는 요청하는 클라이언트나 요청하는 프록시 서버로 반환되거나 전달된다. 그런 다음, 요청된 객체가 사전 인출 요청이면, 요청된 객체는 프록시 서버(24)의 제 1 사전 인출 캐쉬(296)로부터 삭제가능한 것으로 표시된다. 요청된 객체에 대해 사전 인출 요청이 제공되지 않으면, 사전 인출 조회 메시지가 컨텐트 서버(20)로 전달된다. 그러나, 사전 인출 조회 메시지는 요청된 객체가 제 1 사전 인출 캐쉬(296)로부터 전달된 때에만 전달된다. 요청된 객체가 제 1 사전 인출 캐쉬(296) 내에도 존재하지 않고, 제 1 주된 캐쉬(294) 내에도 존재하지 않으면, 객체 요청이 다른 프록시 서버(24)나 컨텐트 서버(20)로 전달된다. 제 1 주된 캐쉬(294) 내에서 발견된 요청된 객체는 요청하는 클라이언트나 프록시 서버(24)로 반환되거나 전달되고, 요청이 사전 인출 요청이 아니었으면 제 1 통계 테이블(293)이 갱신된다.

판단 부호(1105)에서, 요청된 객체 O가 사전 인출 캐쉬 내에서 발견되면, 객체 O는 블럭(1130)에서 요청하는 노드로 반환된다. 판단 부호(1135)에서, 요청이 사전 인출 요청이면, 제 1 사전 인출 캐쉬(296) 내의 객체 O는 블럭(1150)에서 삭제가능한 것으로 표시된다. 제 1 사전 인출 캐쉬(296)로부터 객체를 삭제하면, 사전 인출 객체 O가 이제는 프록시 서버(24)의 하위 계층에 있는 소정 노드에 의해 사전 인출된 것이므로, 장래의 사전 인출을 위한 공간을 만들게 된다. 그렇지 않으면, 객체 O가 실제로 조회된 것이므로, 조회 블럭(1140)에서, 캐쉬 관리자가 호출되어 객체 O를 제 1 사전 인출 캐쉬(296)로부터 제 1 주된 캐쉬(294)로 이동시킨다. 블럭(1160)에서는, 사전 인출 조회 메시지가 프록시 계층 서버를 통해 컨텐트 서버(20)로 전달된다. 블럭(1170)에서는, 갱신 통계 루틴이 호출되어, 사전 인출 객체 O의 실제 조회를 포착하도록 지역 제 1 통계 테이블(293)을 갱신한다. 판단 부호(1110)에서, 객체 O가 제 1 주된 캐쉬(294) 내에서 발견되지 않으면, 조회 블럭(1125)에서 객체 요청이 프록시 서버 계층을 통해 컨텐트 서버(20)로 전달된다. 판단 부호(1115)에서, 객체 요청이 사전 인출 요청이 아니면, 블럭(1120)에서 제 1 갱신 통계 루틴이 호출되어, 객체의 실제 조회를 포착하도록 지역 제 1 통계 테이블(293)을 갱신한다. 블럭(1180)에서, 객체 O는 요청 노드로 반환된다.

도 12는 사전 인출 조회 메시지 처리기(285)의 동작의 예를 도시하고 있다. 사전 인출 조회 메시지 처리기(285)는 먼저 객체가 제 1 사전 인출 캐쉬(296) 내에 있는지 여부를 조사한다. 객체가 제 1 사전 인출 캐쉬(296) 내에 있으면, 사전 인출 조회 메시지를 전달하기 이전에 캐쉬 관리자 루틴이 호출된다. 객체가 제 1 사전 인출 캐쉬(296) 내에 존재하지 않으면, 사전 인출 조회 메시지가 전달된다.

단계(1205)에서 시작하여, 제 1 갱신 통계 루틴(264)이 호출되어, 사전 인출 객체(O)의 실제 조회를 포착하도록 지역 제 1 테이블(293)을 갱신한다. 판단 부호(1210)에서, 객체 O가 제 1 사전 인출 캐쉬(296) 내에 존재하면, 객체가 실제로 조회된 것이므로, 블럭(1215)에서 캐쉬 관리자가 호출되어 객체 O를 제 1 사전 인출 캐쉬(296)로부터 제 1 주된 캐쉬(294)로 이동시킨다. 블럭(1220)에서, 사전 인출 조회 메시지는 다음 상위 계층 프록시로 전달된다.

도 13은 프록시 객체 처리기(284)의 동작의 예를 도시하고 있다. 프록시 객체 처리기(284)는 예외 없이 항상 사전 인출 갱신 유닛(291)이나 사전 인출 힌트 정보 갱신 루틴을 먼저 호출한다. 사전 인출 객체 요청은 사전 인출 객체가 하위 레벨 서버에 의해 요청되었는지 여부에 따라 달리 처리된다. 사전 인출 객체가 하위 레벨 서버에 의해 요청되었으면, 그 객체는 하위 레벨 서버로 반환되고, 그렇지 않으면, 사전 인출 캐쉬 관리자가 호출된다. 사전 인출 객체가 요청되지 않으면, 그 객체는 요청하는 프록시 서버(24)나 클라이언트에게 반환되고, 제 1 갱신 통계 루틴(264)이 호출된다.

조회 블럭(1305)에서 시작하여, PHI 갱신 루틴이나 사전 인출 갱신 수단(291)이 호출되어 객체 O의 PHI를 갱신한다. 조회 블럭(1310)에서, 수신된 객체가 사전 인출 객체가 아니면, 제 1 사전 인출 처리기(248)가 단계(1315)에서 호출된다. 조회 블럭(1320)에서는, 객체가 (하위 계층에 있는) 요청하는 노드로 반환된다. PICS 프로토콜을 이용하는 바람직한 실시예 하에서, (식별자 카테고리 값으로 표시된) 대응 사전 인출 후보가 이 프록시 서버(24)에 의해 사전 인출되면, P 라벨 내의 캐싱 카테고리 값이 갱신된다. 블럭(1325)에서는, 캐쉬 관리자가 호출되어 주된 캐쉬 내에 객체를 저장한다. 조회 블럭(1355)에서는, 갱신 통계 루틴이 호출된다. 블럭(1330)에서, 수신된 사전 인출 객체가 계층 내의 하위 레벨 노드에 의해 요청되면, 객체는 조회 블럭(1350)에서 요청자에게 반환된다. 그렇지 않으면, 객체는 현재 노드에 의해 요청된 것이고, 사전 인출 캐쉬 관리자가 조회 블럭(1340)에서 호출되어, 객체를 위한 공간을 만들기 위해 필요한 교체(replacement)를 함으로써 객체를 사전 인출 캐쉬 내에 저장한다.

도 14는 사전 인출 갱신 수단(291) 또는 PHI 갱신 루틴의 예를 도시하고 있다. PHI 갱신 루틴은 요청된 객체의 지역 카운트를 참조 카운트와 비교한다. 요청된 객체가 제 1 최소 참조값을 초과하면, 제 2 최소 참조값 테스트를 받게 된다. 참조값은 실험 결과에 기초하여 설정될 수 있다. 지역 카운트가 제 1 최소 참조값보다 작으면, 원래 사전 인출 힌트 정보 리스트가 유지된다. 지역 카운트가 제 1 참조값보다 크면서 제 2 참조값보다는 크지 않으면, 사전 인출 힌트 정보 리스트가 수정된다.

판단 부호(1405)에서 시작하여, 지역 제 1 통계 테이블(293) 내의 객체(O)의 지역 카운트(즉, TCount)가 제 1 최소 참조값(즉, CTM)보다 크지 않으면, 객체 O의 헤더 내의 원래 PHI가 유지된다. 이는 프록시 서버(24)의 하위 계층 내의 객체가 임의의 갱신 동작을 보장할 만큼 충분히 조회하지 않았기 때문이다. 판단 부호(1415)에서, 지역 통계 테이블(261) 내의 객체(O)의 지역 카운트(즉, TCount)가 제 2 최소 참조값(즉, TCH)보다 크면, 객체 헤더 내의 원래 PHI는 무시되고, PHI 발생 루틴이 블럭(1420)에서 호출되어, 지역 제 1 통계 테이블(293)에 기초하여 PHI를 발생시킨다. 이는 지역 습성에 기초하여 PHI를 발생시키기 위해 프록시 서버(24)의 하위 계층 내에 있는 객체에 대한 충분한 조회가 있기 때문이다. 그렇지 않으면, 조회 블럭(1425)에서 PHI 수정기가 호출되어, 각 객체에 대해 조회 확률의 지역 추정치와 PHI 리스트 내의 (상위 계층에 의해 설정된) 헤더 내의 값의 평균을 취함으로써 확률 카테고리 값을 수정한다.

프록시 서버(24) 중 일부가 제 1 통계 테이블(293)을 유지하지 않고 지역 조회 패턴을 추적하지 않더라도,

사전 인출 방안을 제 역할을 한다. 비추적(non-tracking) 프록시 서버는 상위 레벨 프록시 서버나 컨텐트 서버(20)로부터 전달된 PHI 내의 확률(즉, PROB) 카테고리 값을 단순히 수용한다.

또한, 사전 인출 방안은 이종(heterogeneous) 프록시 서버 환경에서도 동작하는데, 여기서 프록시 서버의 일부는 사전 인출 프로토콜을 이해하지 못하고 협동 작업에 참여하지 않는 통상적인 프록시 서버이다. 이들 통상적인 프록시 서버는 객체 헤더를 해석하여 사전 인출 동작을 개시하지도 않을 것이며, 제 1 통계 테이블(293)을 유지하지도 않을 것이다. 통상적인 프록시 서버는 객체를 지역적으로 캐싱하기로 결정하면, 메타 데이터(예컨대, 헤더 내의 PICS 라벨)만을 객체와 함께 다음 계층 프록시 서버로 전달하거나, 메타 데이터를 객체와 함께 저장할 것이다.

프록시 계층이 없어도, 본 발명은 임의의 네트워크나 클라이언트 서버 환경에 적용할 수 있다. 객체 송신기 노드는 조회 패턴을 추적하고, PHI를 유도하며, 이를 객체 헤더 내에 포함시킬 수 있다. 또한, 객체 수신기 노드는 PHI를 자신의 캐쉬 내에 객체와 함께 저장할 수 있다. 객체 조회 시, 저장된 PHI, 캐쉬 크기와 전술한 액세스 비용과 같은 지역적 인자에 기초하여 사전 인출 동작이 개시될 수 있다.

PHI를 캐싱된 객체와 함께 저장하면, 사전 인출을 위한 추가적인 기회를 제공한다. 보다 많은 저장이 가능해지거나, 추가 조회가 발생하거나, 사전 인출 객체가 수신되는 것과 같은 이벤트는 캐싱된 객체의 PHI를 조사함으로써 추가적인 사전 인출을 트리거하는 데 사용될 수 있다. 사전 인출 동작은 송신기 개시 접근 방법에서처럼 객체가 서버로부터 요청되는 경우에만 제한되지 않는다.

바람직한 실시예에는, 웹과 프록시 서버를 위한 일반적인 사전 인출 방안이 설명되었지만, 당업자는 본 발명이 사전 인출될 객체가 유사한 특성을 갖는 임의의 상황에 적용가능하며, 인터넷이나 WWW 애플리케이션에 제한될 필요가 없다는 것을 이해할 것이다.

또한, 본 발명의 바람직한 실시예는 계층 내의 부모 및 자녀 프록시 사이의 협동 작업인 사전 인출을 위한 것이지만, 형제 프록시(sibling proxies) 사이의 협동 작업을 포함하도록 용이하게 변형될 수 있다. 예를 들어, 상위 계층 내의 프록시가 요청된 객체나 사전 인출 후보를 캐싱하지 않았으면, 프록시는 형제 프록시에게 문의할 수도 있다.

(57) 청구의 범위

청구항 1

컨텐트 서버(a content server)(20)로부터 선택 정보(select information)를 사전 인출(prefetch)하여 통신 네트워크(25)와 하나 이상의 프록시 서버(24)를 통해 상기 컨텐트 서버와 연관된 하나 이상의 클라이언트(22)에 대해 선택 정보가 용이하게 액세스될 수 있도록 하는 방법에 있어서, 상기 방법은

상기 프록시 서버 중 적어도 하나로부터 정보를 요청하는 단계(100)와,

상기 클라이언트 중 어느 하나에 의한 이전 인출 또는 요청된 정보로의 액세스에 기초하여 상기 클라이언트를 위한 사전 인출 힌트 정보(prefetch hint information)를 설정하는 단계(102, 104)와,

상기 사전 인출 힌트 정보를 이용하여 상기 요청된 정보에 주석(annotate)을 다는 단계(105)와,

상기 사전 인출 힌트 정보 및 상기 요청 정보에 기초하여 상기 서버 중 어느 하나로부터 선택 정보를 사전 인출하는 단계(108)

를 포함하는 선택 정보 사전 인출 방법.

청구항 2

제 1 항에 있어서,

상기 요청된 정보는 요청된 객체를 포함하며, 상기 선택 정보는 사전 인출 후보 객체를 포함하는 선택 정보 사전 인출 방법.

청구항 3

제 2 항에 있어서,

상기 사전 인출 힌트 정보 설정 단계는

상기 클라이언트와 관련된 데이터 조회 패턴(data reference patterns)을 추적하는 단계와,

상기 요청된 객체와 상기 사전 인출 후보 객체 사이의 관계에 따라 상기 데이터 조회 패턴을 객체 식별자 그룹으로 조직하는 단계와,

상기 그룹 중 어느 것이 상기 사전 인출 후보 객체와 상기 요청된 객체 사이에서 선택 정보로서 상기 사전 인출 후보 객체의 사전 인출을 보장하기에 충분한 관계를 갖는지를 결정하는 단계

를 포함하는 선택 정보 사전 인출 방법.

청구항 4

제 3 항에 있어서,

상기 데이터 조회 패턴 추적 및 조직 단계는

조회 데이터 패턴을 프록시 서버에 저장하는 데이터베이스를 설정하는 단계와,

상기 각각의 데이터베이스 내에 필드를 배열하되, 상기 필드는 객체 식별자와 객체 카운트(object counts)를 포함하고, 상기 객체 식별자는 요청된 객체를 서로 구별하며, 상기 객체 카운트는 상기 요청된 객체

35-16

가 상기 서버 중 하나에 의해 요청된 회수를 나타내는 단계와,

상기 데이터베이스와 액세스 리스트(access lists)를 결합하되, 상기 각각의 액세스 리스트는 각각의 요청된 객체와 연관된 하나 이상의 사전 인출 후보 객체를 포함하는 단계

를 포함하는 선택 정보 사전 인출 방법.

청구항 5

제 3 또는 4 항에 있어서,

상기 데이터 조회 패턴 추적 및 조직 단계는

요청된 객체와 사전 인출 후보 객체 사이의 관계를 규정하는 데이터베이스를 설정하는 단계와,

상기 요청된 객체와 상기 사전 인출 후보 객체 상의 액세스 통계(access statistics)를 추적하는 단계를 더 포함하되,

상기 액세스 통계는 요청된 객체의 부모 객체(a parent-object) 액세스 빈도와 사전 인출 후보 객체의 자녀 객체(a childOobject) 액세스 빈도를 포함하고, 상기 부모 객체 액세스 빈도는 상기 요청된 객체 중 특정한 하나가 상기 클라이언트 중 적어도 하나에 의해 요청된 회수를 나타내고, 상기 자녀 객체 액세스 빈도는 상기 특정한 하나의 액세스 바로 다음에 상기 사전 인출 후보 객체 중 하나 이상이 실제로 요청된 회수를 나타내는

선택 정보 사전 인출 방법.

청구항 6

제 5 항에 있어서,

상기 자녀 객체 액세스 빈도와 연관된 상기 부모 객체 액세스 빈도에 대한 상기 자녀 객체 액세스 빈도의 비율에 기초하여 사전 인출 후보 객체의 사전 인출 리스트를 선택하되, 상기 사전 인출 리스트는 참조 확률을 초과하는 비율을 갖는 사전 인출 후보 객체만을 포함하는 단계를 더 포함하는 선택 정보 사전 인출 방법.

청구항 7

제 2 내지 6 항 중 어느 한 항에 있어서,

상기 선택 정보 사전 인출 단계는

상기 프록시 서버 중 하나 또는 상기 클라이언트 중 하나에서 수신한 사전 인출 후보 객체의 서브셋(a subset)을 사전 인출하는 단계와,

상기 프록시 서버 중 하나에서 이용가능한 저장 크기에 기초하여 선택 정보로서 상기 사전 인출 후보 객체의 서브셋을 선택하는 단계

를 포함하는 선택 정보 사전 인출 방법.

청구항 8

제 2 내지 7 항 중 어느 한 항에 있어서,

상기 선택 정보 사전 인출 단계는

상기 프록시 서버 중 하나 또는 상기 클라이언트 중 하나에서 수신한 사전 인출 후보 객체의 서브셋을 사전 인출하는 단계와,

상기 프록시 서버 중 하나에서의 액세스 시간에 기초하여 선택 정보로서 상기 사전 인출 후보 객체의 서브셋을 선택하는 단계

를 포함하는 선택 정보 사전 인출 방법.

청구항 9

제 2 내지 8 항 중 어느 한 항에 있어서,

상기 선택 객체와 상기 요청된 객체는 인터넷 데이터를 포함하고, 상기 요청된 객체는 상기 인터넷 데이터와 연관된 HTTP(hypertext transfer protocol)을 통해 상기 선택 객체를 조회하며, 상기 프록시 서버는 상기 요청된 객체의 객체 헤더에 저장된 요청된 객체 URL(universal resource locator)의 HTTP 프로토콜을 통해 상기 요청된 객체에 대한 상기 선택 객체의 관계를 식별하는 선택 정보 사전 인출 방법.

청구항 10

제 2 내지 9 항 중 어느 한 항에 있어서,

상기 요청된 정보에 주석을 다는 단계는 인터넷 컨텐트 선택을 위한 플랫폼(a Platform for Internet Content Selection : PICS) 포맷, 또는 PICS 포맷과 호환가능한 라벨 프로토콜(a label protocol)을 갖는 사전 인출 라벨 내에 상기 사전 인출 힌트 정보를 배치하는 단계를 포함하는 선택 정보 사전 인출 방법.

청구항 11

제 10 항에 있어서,

상기 PICS 포맷은 객체 식별자 카테고리와 조회 식별자 카테고리(referrer identifier category)를 포함하며,

상기 방법은 상기 조회 식별자 카테고리를 상기 컨텐트 서버로 돌려보내, 실제로 조회된 임의의 사전 인출 후보 객체를 식별하는 단계를 더 포함하는 선택 정보 사전 인출 방법.

청구항 12

제 1 또는 2 항에 있어서,

상기 사전 인출 힌트 정보를 설정하는 단계는

상기 요청된 정보의 이전 인출 또는 이전 액세스에 기초하여 상기 클라이언트의 데이터 조회 패턴을 식별하는 단계와,

상기 데이터 조회 패턴으로부터 사전 인출 힌트 정보를 규정하는 단계

를 포함하는 선택 정보 사전 인출 방법.

청구항 13

제 1 내지 12 항 중 어느 한 항에 있어서,

상기 프록시 서버에는 계층이 존재하고, 상기 계층은 송신 노드(a sending node)와 수신 노드(a receiving node)를 포함하며,

상기 방법은 상기 송신 노드로부터 상기 수신 노드로 사전 인출 힌트 정보를 전송하는 단계와,

초기 상태에서 최종 상태로 상기 사전 인출 힌트 정보를 수정하되, 상기 초기 상태는 상기 송신 노드에 의해 결정되고, 상기 최종 상태는 상기 수신 노드에 의해 결정되며, 상기 최종 상태의 상기 사전 인출 힌트 정보 궁극적인 사전 인출 결정으로서 지정되는 선택 정보 사전 인출 방법.

청구항 14

제 13 항에 있어서,

상기 정보 사전 인출단계는 상기 수신 노드에 의해 개시되는 선택 정보 사전 인출 방법.

청구항 15

제 1 내지 12 항 중 어느 한 항에 있어서,

상기 프록시 서버를 서버의 계층으로 조직하되, 상기 계층은 다수 계층의 프록시 서버를 구비하고, 상기 다수 계층은 상기 네트워크에 대한 상기 프록시 서버 각각의 근접도(proximity)에 기초하여 결정되며, 최고 계층은 상기 네트워크에 가장 근접한 단계와,

상기 프록시 서버로부터 클라이언트로 상기 계층을 통해 전송된 상기 사전인출 힌트 정보를 해석하고 갱신하는 단계

를 더 포함하는 선택 정보 사전 인출 방법.

청구항 16

제 15 항에 있어서,

프록시 서버는 상기 요청된 정보에 대한 지역 조회(local references)가 존재하고 상기 지역 계층 서버의 카운트가 최소 참조 카운트를 초과하면 상기 사전 인출 힌트 정보를 갱신하는 선택 정보 사전 인출 방법.

청구항 17

제 15 또는 16 항에 있어서,

상기 사전 인출 힌트 정보는 상기 프록시 계층 내의 캐쉬 상태(caching status)에 따라 갱신되는 선택 정보 사전 인출 방법.

청구항 18

네트워크(25)를 통해 컨텐트 서버(20)로부터 선택 정보를 사전 인출하는 데이터 처리 시스템에 있어서, 상기 시스템은 상기 네트워크와 연관된 하나 이상의 프록시 서버(24), 상기 컨텐트 서버와 하나 이상의 상기 프록시 서버와 상기 네트워크를 통해 통신하는 클라이언트를 포함하며, 상기 시스템은

상기 프록시 서버 중 적어도 하나로부터 정보를 요청하는 수단과,

상기 클라이언트 중 어느 하나에 의한 이전 인출 또는 요청된 정보로의 액세스에 기초하여 상기 클라이언트를 위한 사전 인출 힌트 정보를 설정하는 수단과,

상기 사전 인출 힌트 정보를 이용하여 상기 요청된 정보에 주석을 다는 수단과,

상기 사전 인출 힌트 정보 및 상기 요청 정보에 기초하여 상기 서버 중 어느 하나로부터 선택 정보를 사전 인출하는 수단

을 포함하는 데이터 프로세싱 시스템.

청구항 19

제 18 항에 있어서,

상기 요청된 정보는 요청된 객체를 포함하며, 상기 선택 정보는 사전 인출 후보 객체를 포함하고, 상기 각각의 서버와 상기 클라이언트 중 적어도 하나는 주 캐쉬와 사전 인출 캐쉬 사이에 적어도 논리적 구별(logical separation)을 갖는 메모리를 구비하며, 상기 사전 인출 선택 정보는 상기 사전 인출 캐쉬에 저장되는 데이터 처리 시스템.

청구항 20

제 19 항에 있어서,

상기 사전 인출 수단은 객체를 사전 인출하고 상기 사전 인출된 객체를 우선 순위 방안(a priority scheme)에 기초하여 상기 사전 인출 캐쉬에 저장하되, 상기 우선 순위 방안은 최고 사전 인출 값을 갖는 객체를 상기 사전 인출 캐쉬의 이용가능한 메모리에 대해 먼저 저장하고, 상기 사전 인출 값은 하나 이상의 클라이언트에 의한 상기 사전 인출 객체에 대한 개별적이거나 통합된 조회 가능성(reference probability)에 기초하는 데이터 처리 시스템.

청구항 21

제 20 항에 있어서,

상기 사전 인출 값은 상기 사전 인출 힌트 정보 리스트 상의 사전 인출 후보 객체가 클라이언트 중 특정한 하나에 의해 액세스 될 추정된 액세스 확률과 상기 사전 인출 후보 객체 중 선택된 하나를 저장하는 데 필요한 저장 크기에 기초한 상기 객체의 액세스 비용에 기초하는 데이터 처리 시스템.

청구항 22

제 20 또는 21 항에 있어서,

상기 사전 인출 수단은 상기 사전 인출 값을 계산하는 계산 수단과 상기 계산 수단에 의해 계산된 상기 사전 인출 값을 평가하는 평가 수단을 포함하는 사전 인출 처리기를 포함하고, 상기 계산 수단은 상기 사전 인출 힌트 정보 리스트 상의 각각의 사전 인출 객체 식별자에 대해 상기 사전 인출 값을 계산하고, 상기 평가 수단은 다수의 후보 사전 인출 객체의 사전 인출 값을 평가하고 상기 사전 인출 힌트 정보 리스트 상의 최고 사전 인출 값과 연관된 후보 사전 인출 객체의 캐쉬의 우선 순위를 매기는 데이터 처리 시스템.

청구항 23

제 22 항에 있어서,

상기 사전 인출 수단은 상기 프록시 서버 중 적어도 하나 안에 제 1 사전 인출 처리기를, 상기 클라이언트 중 적어도 하나 안에 제 2 사전 인출 처리기를 포함하는 데이터 처리 시스템.

청구항 24

제 19 내지 23 항 중 어느 한 항에 있어서,

상기 클라이언트 중 하나의 메모리에 저장된 클라이언트 객체 요청 처리기를 포함하되, 상기 클라이언트 객체 요청 처리기는 프록시 서버나 상기 네트워크로부터 상기 요청된 객체를 요청하기 이전에 요청된 객체를 위한 주된 캐쉬(main cache)나 사전 인출 캐쉬를 검색하고, 상기 클라이언트 객체 요청 처리기는 상기 객체가 상기 사전 인출 캐쉬 내에 있는 경우, 상기 사전 인출 객체가 상기 클라이언트에 의해 실제로 액세스되었음을 나타내는 사전 인출 조회 메시지(a prefetch referenced message)를 상기 서버로 발생시킴으로써 상기 컨텐트 서버나 상기 프록시 서버로 통지하는 데이터 처리 시스템.

청구항 25

제 19 내지 24 항 중 어느 한 항에 있어서,

상기 프록시 서버 중 적어도 하나는 요청된 객체가 상기 사전 인출 캐쉬 내에 위치하는 경우 캐쉬 관리자 루틴(a cache manager routine)을 호출한 후 사전 인출 조회 메시지를 전달하는 사전 인출 조회 메시지 처리기를 포함하는 데이터 처리 시스템.

청구항 26

제 18 내지 25 항 중 어느 한 항에 있어서,

상기 요청된 정보는 요청된 객체를 포함하고, 상기 선택 정보는 사전 인출 후보 객체를 포함하며, 상기 컨텐트 서버 또는 상기 프록시 서버는

부모 객체 식별자와 연관된 자녀 객체 식별자를 저장하는 통계 테이블과,

상기 통계 테이블 내의 정보에 기초하여 상기 사전 인출 힌트 정보 리스트를 발생시키는 발생 수단을 포함하되,

상기 통계 테이블은 부모 객체의 제 1 카운트를 포함하고, 상기 제 1 카운트는 부모 객체가 상기 클라이언트에 의해 조회된 빈도를 나타내며, 상기 통계 테이블은 자녀 객체의 제 2 카운트를 포함하고, 상기 제 2 카운트는 상기 대응 자녀 객체가 상기 부모 객체 중 선택된 하나가 조회된 후 조회되는 빈도를 나타내는 데이터 처리 시스템.

청구항 27

제 26 항에 있어서,

상기 발생 수단은 상기 제 1 카운트와 상기 대응하는 제 2 카운트의 입력을 수용하여 상기 사전 인출 힌트 정보 리스트를 유도하고, 상기 발생 수단은 상기 각각의 부모 객체 식별자에 대해 상기 제 1 카운트에 대한 상기 제 2 카운트의 비율을 계산하는 계산 수단과, 상기 비율을 힌트 참조값과 비교하는 비교 수단을 포함하되, 상기 비교 수단은 상기 비율이 상기 힌트 참조값을 초과하는 경우 상기 사전 인출 정보 리스트 상의 자녀 객체와 연관된 부모 객체를 그룹화하거나 연합하는 데이터 처리 시스템.

청구항 28

제 26 항에 있어서,

상기 발생 수단은 통계 테이블로부터 제 1 카운트, 제 2 카운트, 자녀 객체 식별자, 부모 객체 식별자의 입력을 수용하고, 상기 발생 수단은 부모 객체와 그들의 연관된 자녀 객체 사이의 액세스 관계를 평가하는 평가 수단을 포함하되, 상기 평가 수단은 상기 자녀 객체가 대응 부모 객체에 대한 클라이언트에 의한 조회 이전에, 바로 다음 또는 후속적으로 적절한 빈도로 조회되는 경우, 사전 인출 힌트 정보 리스트 상에 자녀 객체 식별자를 배치하는 데이터 처리 시스템.

청구항 29

제 28 항에 있어서,

상기 발생 수단은 대응하는 부모 액세스에 대한 자녀 액세스의 비율에 기초하여 상기 적절한 빈도를 계산하는 계산 수단을 더 포함하는 데이터 처리 시스템.

청구항 30

제 26 내지 29 항 중 어느 한 항에 있어서,

상기 서버는 실제로 요청되는 자녀 객체에 기초하여 상기 통계 테이블을 갱신하는 통계 갱신 수단을 더 포함하되, 상기 통계 갱신 수단은 상기 부모 객체 중 조회하는 하나(a referring one)의 신원을 먼저 검색하되, 상기 조회하는 하나, 즉 조회자 객체(referrer object)는 자녀 객체가 실제로 요청되기 이전에 조회되며, 상기 자녀 객체는 자신의 대응 제 2 카운트를 상기 통계 테이블 내에서 갱신시키면서 실제로 요청되는 데이터 처리 시스템.

청구항 31

제 30 항에 있어서,

상기 통계 갱신 수단은 사전 인출 조회 메시지 처리기로부터 사전 인출 조회 메시지를 수신하면 상기 통계 테이블을 갱신하는 데이터 처리 시스템.

청구항 32

제 26 내지 31 항 중 어느 한 항에 있어서,

상기 서버는 프록시 서버이고 상기 사전 인출 힌트 정보 리스트를 수신하고 갱신하는 사전 인출 갱신 수단을 더 포함하되, 상기 사전 인출 갱신 수단은 제 1 지역 참조 카운트와 상기 제 1 카운트를 비교하고, 상기 사전 인출 갱신 수단은 상기 제 1 카운트가 상기 제 1 지역 참조 카운트를 초과하는 상기 자녀 객체 식별자 각각에 대해 상기 발생 수단을 호출하며, 상기 사전 인출 갱신 수단은 상기 제 1 카운트가 상기 제 1 지역 참조 카운트를 초과하지 않는 경우, 상기 원래 사전 인출 힌트 정보 리스트를 유지하는 데이터 처리 시스템.

청구항 33

제 32 항에 있어서,

상기 사전 인출 갱신 수단은 상기 제 1 카운트를 제 1 및 제 2 지역 참조 카운트와 비교하여, (1) 상기 제 1 카운트가 상기 제 1 지역 참조 카운트를 초과하지 않으면 상기 원래 사전 인출 힌트 정보 리스트를 유지하고, (2) 상기 제 1 카운트가 상기 제 1 지역 참조 카운트를 초과하면서 상기 제 2 지역 참조 카운트를 초과하지 않으면 상기 원래 사전 인출 힌트 정보 리스트를 수정하며, (3) 상기 제 1 카운트가 상기 제 1 및 제 2 지역 참조 카운트를 초과하면 상기 원래 사전 인출 힌트 정보 리스트를 교체하는 데이터 처리 시스템.

청구항 34

제 32 도는 33 항에 있어서,

상기 요청된 객체를 요청한 하위 계층의 프록시 서버나 클라이언트에게로 요청된 객체를 돌려보내는 프록시 객체 처리기를 더 포함하되, 상기 프록시 객체 처리기는 상기 요청된 객체가 사전 인출 객체가 아닌 경우 상기 요청된 객체를 돌려보내고 따라서 상기 통계 테이블을 갱신하는 데이터 처리 시스템.

청구항 35

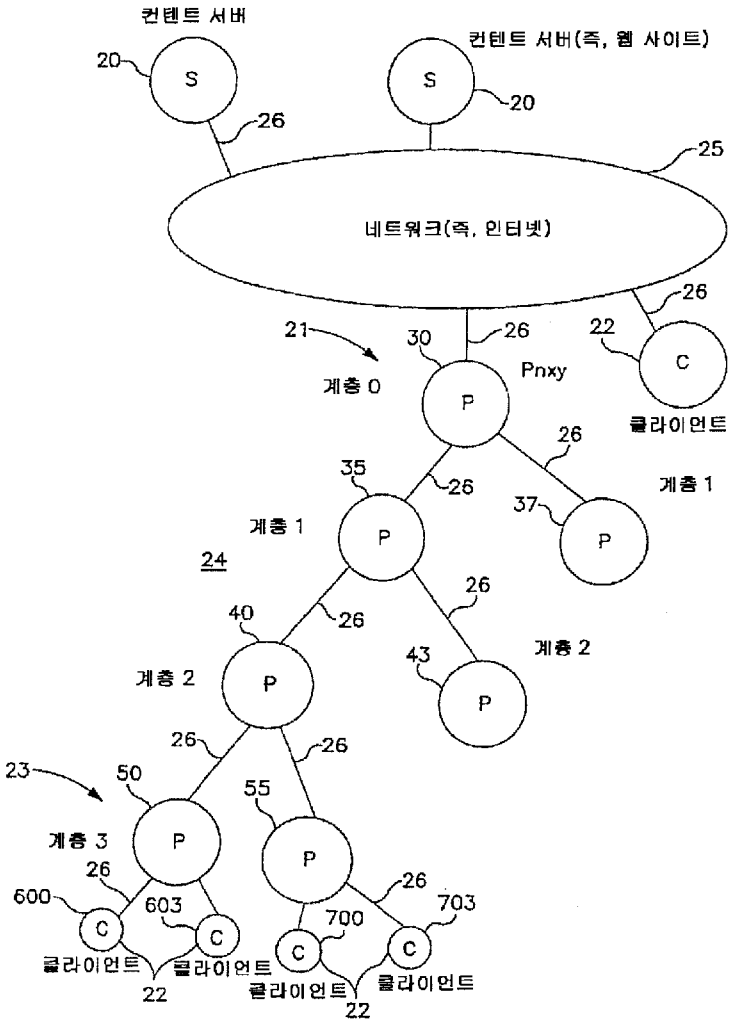
제 34 항에 있어서,

사전 인출 캐쉬를 관리하는 사전 인출 캐쉬 관리자를 더 포함하되, 상기 프록시 객체 처리기는 상기 사전 인출 캐쉬로부터 요청된 객체를 요청 클라이언트나 프록시 서버로 돌려보내고, 상기 객체가 사전 인출 요청에 따라 요청된 경우 상기 사전 인출 캐쉬로부터 삭제가능한 것으로 상기 객체를 표시하고, 상기 프록시 객체 처리기는 상기 요청된 객체가 주된 캐쉬나 사전 인출 캐쉬 내에 있지 않으면 객체 요청을 다른 서버로 보내고, 상기 프록시 객체 처리기는 상기 요청된 객체가 사전 인출 요청과 연관되지 않고 상기 요청된 객체가 상기 사전 인출 캐쉬나 상기 주된 캐쉬 내에 위치하면 상기 통계 테이블을 갱신하는 데이터 처리

시스템.

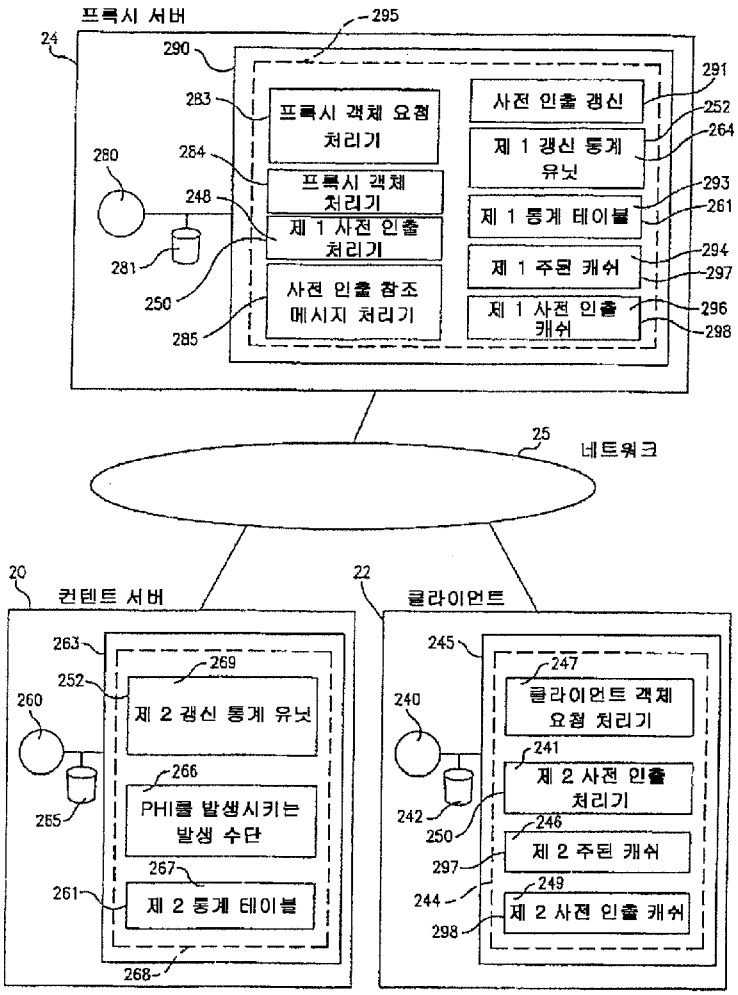
도면

도면1



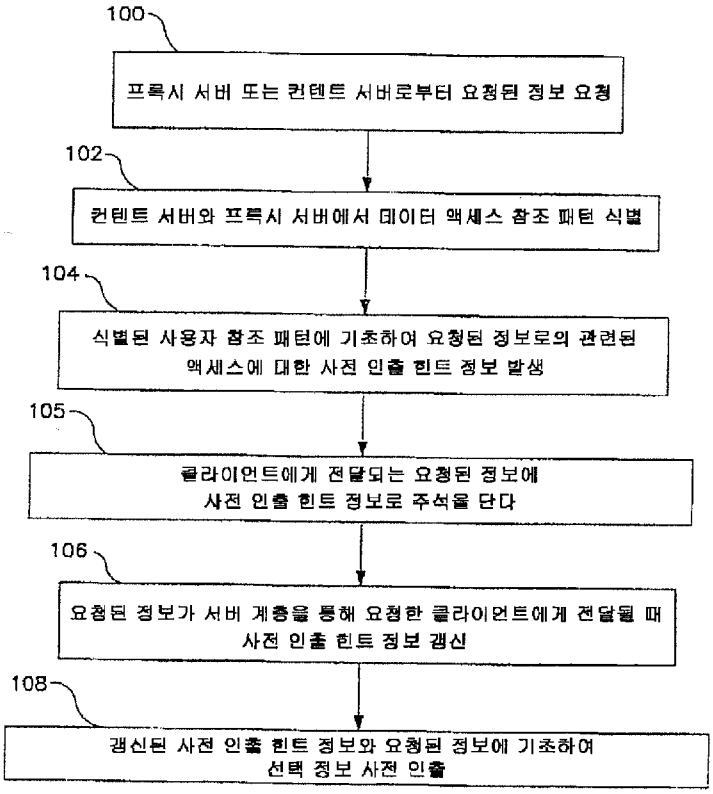
35-21

도면2


프록시 서버
24
295
290
283
프록시 객체 요청 처리기
291
사전 인출 갱신
252
264
제 1 갱신 통계 유닛
280
284
프록시 객체 처리기
293
261
제 1 통계 테이블
248
제 1 사전 인출 처리기
294
제 1 주된 캐쉬
297
281
250
사전 인출 참조 메시지 처리기
296
제 1 사전 인출 캐쉬
298
285
25
네트워크
20
컨텐트 서버
22
클라이언트
263
269
252
제 2 갱신 통계 유닛
260
266
PHI를 발생시키는 발생 수단
265
267
261
제 2 통계 테이블
268
245
247
클라이언트 객체 요청 처리기
240
241
제 2 사전 인출 처리기
242
250
246
제 2 주된 캐쉬
297
244
249
제 2 사전 인출 캐쉬
298

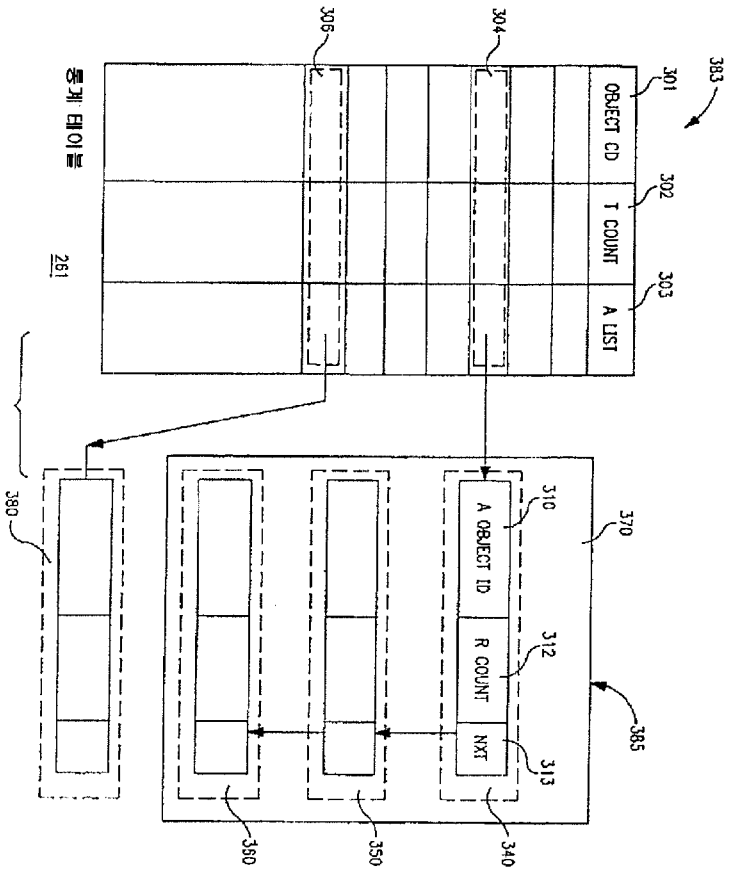
35-22

도면3a


100
프록시 서버 또는 컨텐트 서버로부터 요청된 정보 요청
102
컨텐트 서버와 프록시 서버에서 데이터 액세스 참조 패턴 식별
104
식별된 사용자 참조 패턴에 기초하여 요청된 정보로의 관련된 액세스에 대한 사전 인출 힌트 정보 발생
105
클라이언트에게 전달되는 요청된 정보에 사전 인출 힌트 정보로 주석을 단다
106
요청된 정보가 서버 계층을 통해 요청한 클라이언트에게 전달될 때 사전 인출 힌트 정보 갱신
108
갱신된 사전 인출 힌트 정보와 요청된 정보에 기초하여 선택 정보 사전 인출

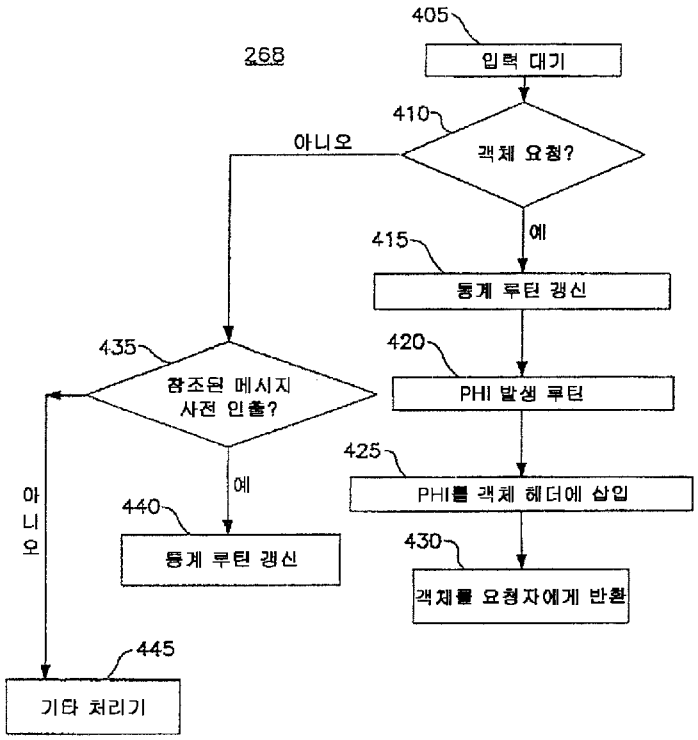
35-23

도면36


중계 테이블
261
OBJECT CD
T COUNT
A LIST
A OBJECT ID
R COUNT
NXT
301
302
303
304
305
310
312
313
340
350
360
370
380
383
385

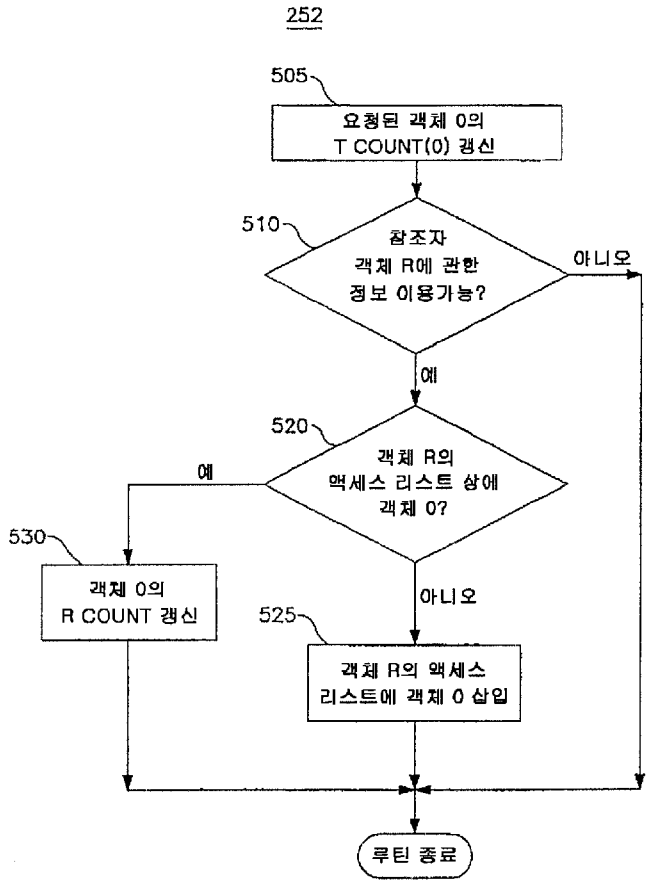
35-24

도면4

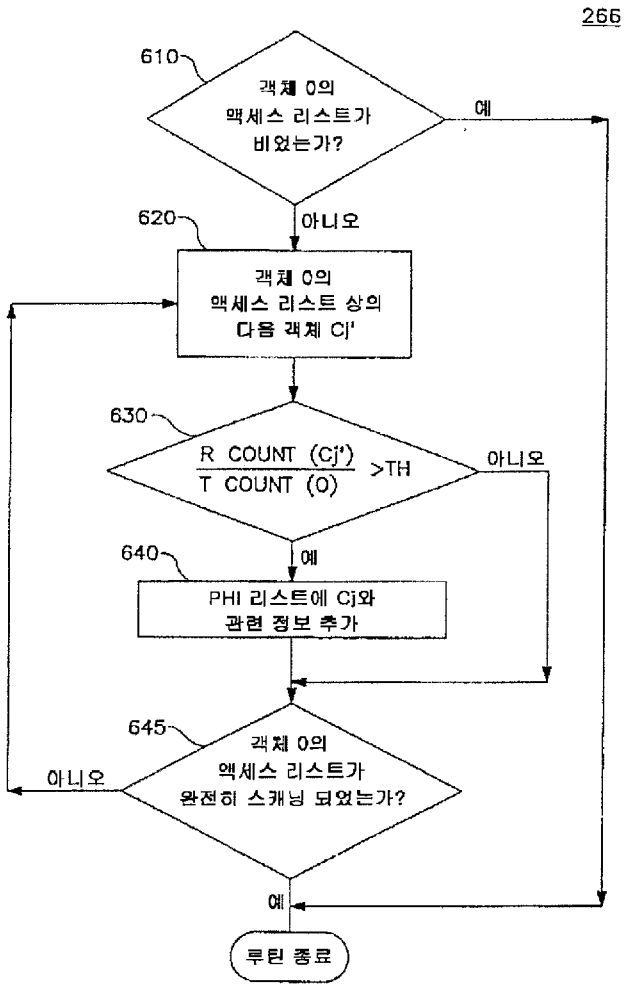

268
405
입력 대기
410
객체 요청?
아니오
예
415
통계 루틴 갱신
420
PHI 발생 루틴
425
PHI를 객체 헤더에 삽입
430
객체를 요청자에게 반환
435
참조된 메시지 사전 인출?
예
아니오
440
통계 루틴 갱신
445
기타 처리기

35-25

도면5

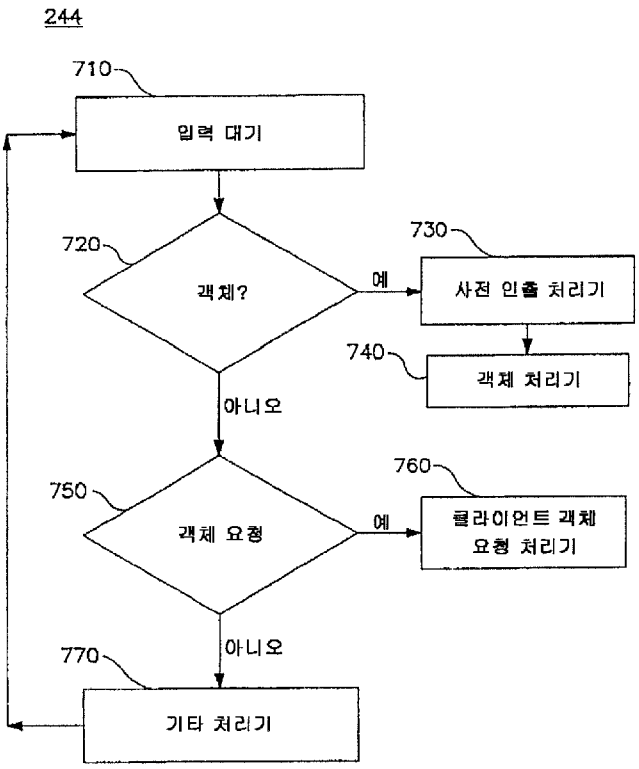

252
505
요청된 객체 O의
T COUNT(O) 갱신
510
참조자
객체 R에 관한
정보 이용가능?
아니오
예
520
객체 R의
액세스 리스트 상에
객체 O?
예
530
객체 O의
R COUNT 갱신
아니오
525
객체 R의 액세스
리스트에 객체 O 삽입
루틴 종료

도면6


265
610
객체 O의
액세스 리스트가
비었는가?
예
아니오
620
객체 O의
액세스 리스트 상의
다음 객체 Cj'
630
R COUNT (Cj')
T COUNT (O)
>TH
아니오
예
640
PHI 리스트에 Cj와
관련 정보 추가
645
객체 O의
액세스 리스트가
완전히 스캐닝 되었는가?
아니오
예
루틴 종료

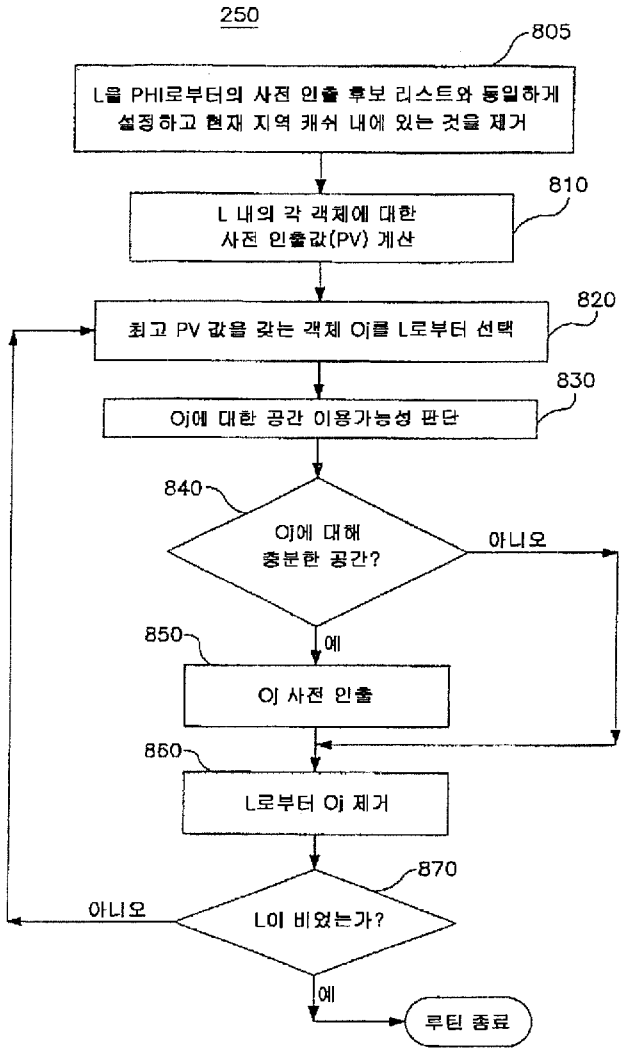
35-27

도면7


244
710
입력 대기
720
객체?
예
730
사전 인출 처리기
740
객체 처리기
아니오
750
객체 요청
예
760
클라이언트 객체 요청 처리기
아니오
770
기타 처리기

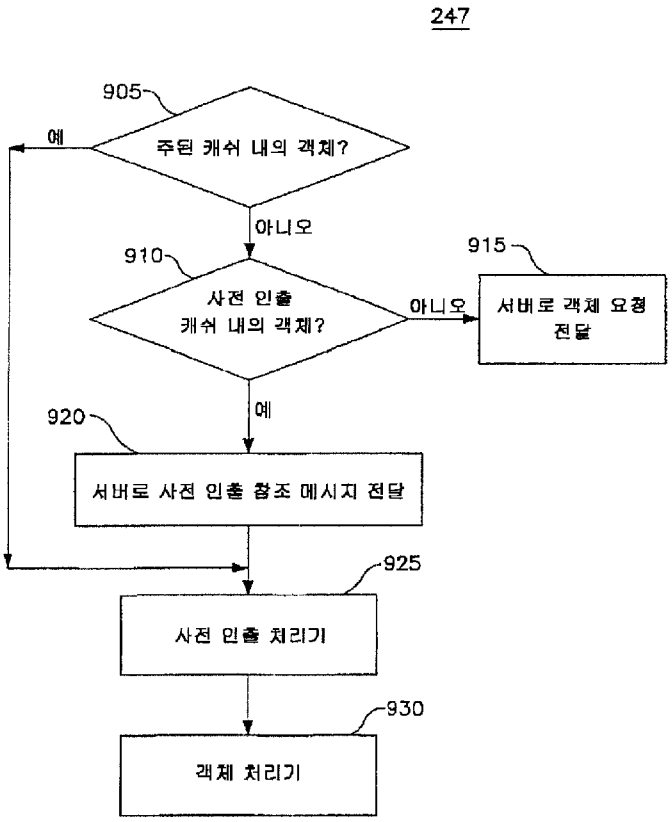
35-28

도면8


250
805
L을 PHI로부터의 사전 인출 후보 리스트와 동일하게 설정하고 현재 지역 캐쉬 내에 있는 것을 제거
810
L 내의 각 객체에 대한 사전 인출값(PV) 계산
820
최고 PV 값을 갖는 객체 Oi를 L로부터 선택
830
Oi에 대한 공간 이용가능성 판단
840
Oi에 대해 충분한 공간?
아니오
예
850
Oi 사전 인출
860
L로부터 Oi 제거
870
L이 비었는가?
아니오
예
루틴 종료

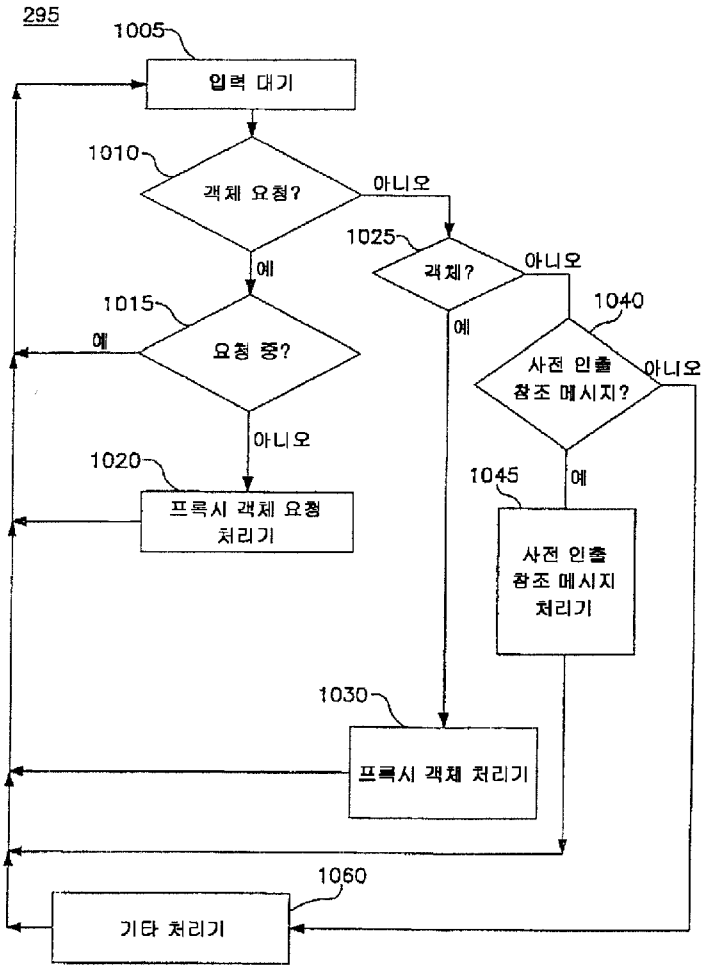
35-29

도면9


247
905
주된 캐쉬 내의 객체?
예
아니오
910
사전 인출
캐쉬 내의 객체?
아니오
915
서버로 객체 요청
전달
예
920
서버로 사전 인출 참조 메시지 전달
925
사전 인출 처리기
930
객체 처리기

35-30

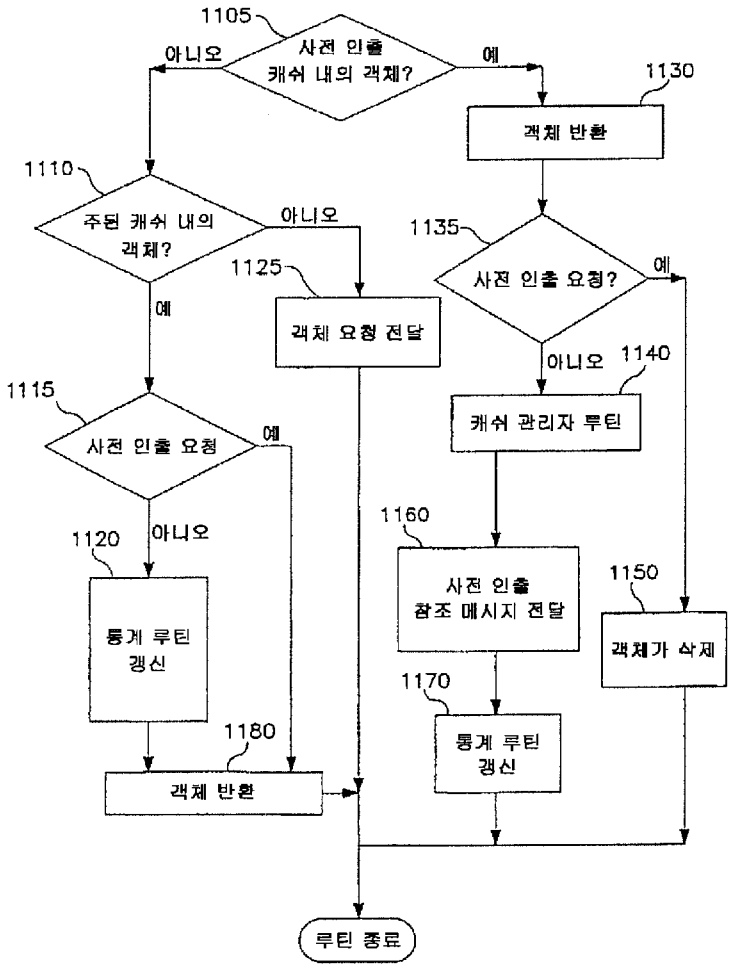
도면10


295
1005
입력 대기
1010
객체 요청?
아니오
예
1025
객체?
아니오
예
1015
요청 중?
예
아니오
1040
사전 인출 참조 메시지?
아니오
예
1020
프록시 객체 요청 처리기
1045
사전 인출 참조 메시지 처리기
1030
프록시 객체 처리기
1060
기타 처리기

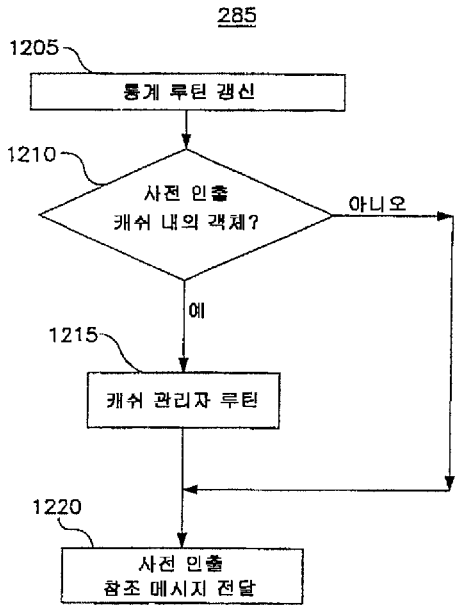
35-31

도면11

283

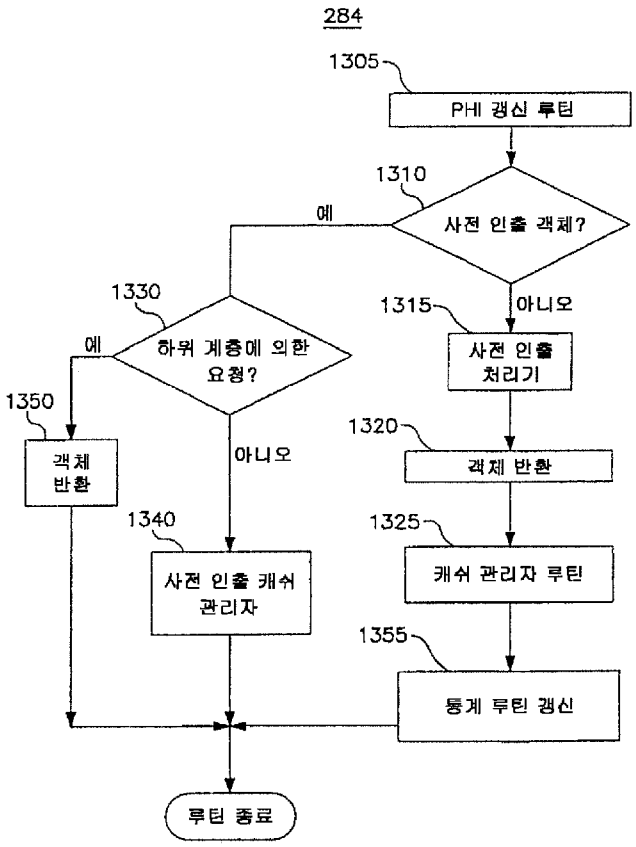

1105
사전 인출
캐쉬 내의 객체?
아니오
예
1130
객체 반환
1110
주된 캐쉬 내의
객체?
아니오
1135
사전 인출 요청?
예
1125
객체 요청 전달
예
아니오
1140
캐쉬 관리자 루틴
1115
사전 인출 요청
예
1160
사전 인출
참조 메시지 전달
1150
객체가 삭제
아니오
1120
통계 루틴
갱신
1170
통계 루틴
갱신
1180
객체 반환
루틴 종료

도면12


285
1205
통계 루틴 갱신
1210
사전 인출
캐쉬 내의 객체?
아니오
예
1215
캐쉬 관리자 루틴
1220
사전 인출
참조 메시지 전달

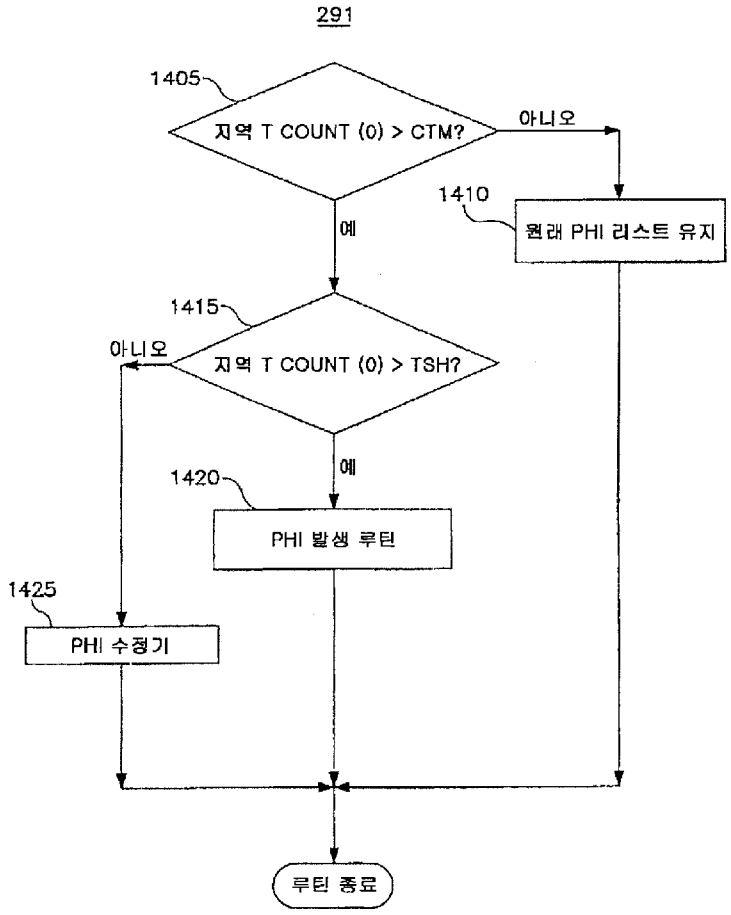
35-33

도면13


284
1305
PHI 갱신 루틴
1310
사전 인출 객체?
예
아니오
1330
하위 계층에 의한 요청?
1315
사전 인출 처리기
예
1350
객체 반환
아니오
1320
객체 반환
1340
사전 인출 캐쉬 관리자
1325
캐쉬 관리자 루틴
1355
통계 루틴 갱신
루틴 종료

35-34

도면14


291
1405
지역 T COUNT (0) > CTM?
아니오
1410
원래 PHI 리스트 유지
예
1415
아니오
지역 T COUNT (0) > TSH?
예
1420
PHI 발생 루틴
1425
PHI 수정기
루틴 종료

35-35

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号
特開2001－7844
（P2001－7844A）
(43)公開日　平成13年１月12日(2001.1.12)

(51)Int.Cl.[7]		識別記号	FI		テーマコード*（参考）	
Ｈ０４Ｌ	12/54		Ｈ０４Ｌ	11/20	１０１Ｂ	５Ｂ０４５
	12/58		Ｇ０６Ｆ	13/00	３５７Ｚ	５Ｂ０８９
Ｇ０６Ｆ	13/00	３５７		15/177	６７４Ａ	５Ｋ０３０
	15/177	６７４	Ｈ０４Ｌ	11/20	１０２Ｄ	
Ｈ０４Ｌ	12/56					

審査請求　未請求　請求項の数162　ＯＬ　（全 46 頁）

(21)出願番号　特願平11－178474

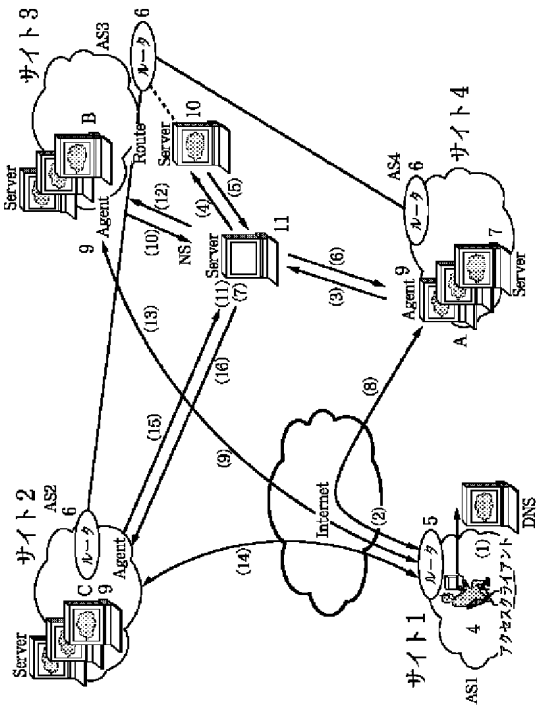
(22)出願日　平成11年６月24日(1999.6.24)

(71)出願人　000001007
キヤノン株式会社
東京都大田区下丸子３丁目30番２号
(72)発明者　荻野　司
神奈川県横浜市港北区新吉田町1268－１－317
(74)代理人　100090538
弁理士　西山　恵三　（外１名）

最終頁に続く

(54)【発明の名称】　ネットワークステータスサーバ及び情報配信システム、及びその制御方法、及びその制御プログラムを格納した記憶媒体

(57)【要約】
【課題】　分散配置されたｗｅｂサーバサイト間のアクセスを平準化するとともに、各々のｗｅｂサーバサイト内のｗｅｂサーバ間においてもアクセス負荷の平準化をする。また、クライアントを最適なｗｅｂサーバにアクセスさせる。
【解決手段】　ネットワークステータスサーバは、クライアントからアクセスされたｗｅｂサーバからの問い合せを受けて、経路情報を用いて最適なｗｅｂサーバを決定してアクセスされたｗｅｂサーバに返答する。返答を受けたｗｅｂサーバは、クライアントに次回からは該最適なｗｅｂサーバにアクセスするよう指示する。



【特許請求の範囲】

【請求項１】　分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得手段と、前記取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定手段とを備えることを特徴とするネットワークステータスサーバ。

【請求項２】　サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得手段は、前記論理的な距離を取得することを特徴とする請求項１記載のネットワークステータスサーバ。

【請求項３】　更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を蓄積する経路情報蓄積手段を備え、前記論理的距離取得手段は、前記経路情報蓄積手段に前記経路情報が蓄積されている場合、前記経路情報蓄積手段に蓄積されている経路情報から、前記論理的な距離を取得することを特徴とする請求項２記載のネットワークステータスサーバ。

【請求項４】　前記サイト決定手段は、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間の各々の論理的な距離が最小となるサイトを前記最適なサイトとして決定することを特徴とする請求項１乃至３のいずれかに記載のネットワークステータスサーバ。

【請求項５】　更に、前記サイト決定手段により決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定するサーバ決定手段を備えることを特徴とする請求項１乃至４のいずれかに記載のネットワークステータスサーバ。

【請求項６】　前記サーバ決定手段は、前記最適なサイト内で予め決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする請求項５記載のネットワークステータスサーバ。

【請求項７】　前記サーバ決定手段は、所定の順序で決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする請求項５記載のネットワークステータスサーバ。

【請求項８】　前記サーバ決定手段は、前記最適なサイト内の情報配信サーバから任意に情報配信サーバを決定することを特徴とする請求項５記載のネットワークステータスサーバ。

【請求項９】　更に、前記分散配置された情報配信サーバを有するサイト内の状態の情報を収集する状態情報収集手段を有し、前記サーバ決定手段は、収集した前記最適なサイト内の状態の情報を条件として、最適な情報配信サーバを決定することを特徴とする請求項５記載のネットワークステータスサーバ。

【請求項１０】　収集した前記サイト内の状態の情報を蓄積する状態情報蓄積手段とを有し、前記状態情報蓄積手段に前記最適なサイト内の状態の情報が蓄積されている場合、前記サーバ決定手段は、蓄積された前記最適なサイト内の状態の情報を条件として、最適な情報配信サーバを決定することを特徴とする請求項９記載のネットワークステータスサーバ。

【請求項１１】　前記状態情報蓄積手段は、前記蓄積したサイト内の状態の情報を所定の期間だけ蓄積することを特徴とする請求項１０に記載のネットワークステータスサーバ。

【請求項１２】　前記状態情報蓄積手段は、前記サイト内の状態の情報を所定の近似式で近似して蓄積することを特徴とする請求項１０又は１１記載のネットワークステータスサーバ。

【請求項１３】　前記サイト内の状態の情報とは、前記サイト内のネットワークの状態の情報と、前記サイト内の情報配信サーバの状態の情報との少なくともどちらか一つの情報であることを特徴とする請求項９乃至１２のいずれかに記載のネットワークステータスサーバ。

【請求項１４】　前記分散配置された情報配信サーバを有する各サイト内のネットワークの状態の情報を収集する場合、前記収集手段は、前記各サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともどれか一つを収集することを特徴とする請求項１３記載のネットワークステータスサーバ。

【請求項１５】　前記分散配置された情報配信サーバの状態の情報を収集する収集手段は、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷の内、少なくともいずれか一つを収集することを特徴とする請求項１３又は１４のいずれかに記載のネットワークステータスサーバ。

【請求項１６】　分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得手段と、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集手段とを有し、前記論理的距離取得手段により取得した論理的な距離と、前記収集手段により収集した前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定するサーバ決定手段とを備えることを特徴とするネットワークステータスサーバ。

【請求項１７】　サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有

するルートサーバから、前記論理的距離取得手段は、前記論理的な距離を取得することを特徴とする請求項１６記載のネットワークステータスサーバ。

【請求項１８】　更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を蓄積する経路情報蓄積手段を備え、前記論理的距離取得手段は、前記経路情報蓄積手段に前記経路情報を蓄積している場合、前記経路情報蓄積手段に蓄積している経路情報から、前記論理的な距離を取得することを特徴とする請求項１７記載のネットワークステータスサーバ。

【請求項１９】　更に、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを蓄積する状態情報蓄積手段を備え、前記収集手段は、前記状態情報蓄積手段に前記状態の情報を蓄積している場合、前記状態情報蓄積手段に蓄積している状態の情報から、前記状態の情報を収集することを特徴とする請求項１６乃至１８のいずれかに記載のネットワークステータスサーバ。

【請求項２０】　前記状態情報蓄積手段は、前記蓄積したサイト内の状態の情報を所定の期間だけ蓄積することを特徴とする請求項１９に記載のネットワークステータスサーバ。

【請求項２１】　前記状態情報蓄積手段は、前記サイト内の状態の情報を所定の近似式で近似して蓄積する請求項１９又は２０のいずれかに記載のネットワークステータスサーバ。

【請求項２２】　前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報を収集する場合、前記収集手段は、前記アクセスしてきたクライアントと前記各サイトとの間の、応答時間、ルータ段数、パケットロス率のうち、少なくともどれか一つを収集することを特徴とする請求項１６乃至２１のいずれかに記載のネットワークステータスサーバ。

【請求項２３】　前記分散配置された情報配信サーバを有するサイト内の状態の情報とは、前記分散配置された情報配信サーバを有するサイト内のネットワークの状態の情報、又は、前記情報配信サーバの状態の情報の少なくともどちらか一つであることを特徴とする請求項１６乃至２２のいずれかに記載のネットワークステータスサーバ。

【請求項２４】　前記分散配置された情報配信サーバを有する各サイト内のネットワークの状態の情報を収集する場合、前記収集手段は、前記各サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともいずれか一つを収集することを特徴とする請求項２３記載のネットワークステータスサーバ。

【請求項２５】　前記分散配置された情報配信サーバの状態の情報を収集する場合、前記収集手段は、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷の内、少なくともいずれか一つを収集することを特徴とする請求項２３又は２４記載のネットワークステータスサーバ。

【請求項２６】　前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を蓄積する経路情報蓄積手段と、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを蓄積する状態情報蓄積手段とを備え、前記サーバ決定手段は、蓄積された前記経路情報と前記状態の情報と所定の重み係数とに基づいて、下記の式；

・ネットワーク状態値：Ｋ１ｎ＝ＲＴｎ・Ａ＋ＲＮｎ・Ｂ＋ＰＬｎ・Ｃ

・サイト内ネットワーク状態値：Ｋ２ｎ＝ＣＳｎ・Ｄ＋ＰＳｎ・Ｅ＋ＥＳｎ・Ｆ

・サーバ状態値：Ｋ３ｎｍ＝ＣＰＵｎｍ・Ｇ＋ＩＤＬＥｎｍ・Ｈ＋ＬＩＮＫｎｍ

・Ｉ＋ＩＯｎｍ・Ｊ

・最適サイト判定値：Ｋｎ＝Ｋ１ｎ・Ｋ＋Ｋ２ｎ・Ｌ＋ＡＳＬｎ・Ｍ

・最適サーバ判定値：Ｓｎｍ＝Ｋｎ・Ｎ＋Ｋ３ｎｍ・Ｏ

（但し、ｎ：サーバサイト番号、ｍ：サーバ番号、Ａ～Ｏ：重み係数であり、各記号はそれぞれ、ＡＳＬｎ：ＡＳパスによる論理的なネットワーク間距離（ＢＧＰの経路情報を用いる）、ＲＴｎ：応答時間、ＲＮｎ：ルータ段数（ルータホップ数）、ＰＬｎ：パケットロス率、ＣＳｎ：サイト内の混雑度、ＰＳｎ：サイト内のパケット数、ＥＳｎ：パケットエラー、ＣＰＵｎｍ：ＣＰＵ負荷率、ＩＤＬＥｎｍ：ＣＰＵアイドル値、ＬＩＮＫｎｍ：コネクションリンク数、ＩＯｎｍ：ディスク負荷である。）における最適サーバ判定値Ｓｎｍが所定の条件を満たす情報配信サーバを最適な情報配信サーバとして決定することを特徴とする請求項１６記載のネットワークステータスサーバ。

【請求項２７】　分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得手段と、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集手段とを有し、前記クライアントからのアクセスが最初のアクセスであると判断した場合、前記論理的距離取得手段により取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイト

を決定するサイト決定手段と、前記サイト決定手段により決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定する第１のサーバ決定手段とを備え、前記クライアントからのアクセスが最初のアクセスでないと判断した場合、前記論理的距離取得手段により取得した論理的な距離と、前記収集手段により収集した前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定する第２のサーバ決定手段とを備えることを特徴とするネットワークステータスサーバ。

【請求項２８】　サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得手段は、前記論理的な距離を取得することを特徴とする請求項２７記載のネットワークステータスサーバ。

【請求項２９】　更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を蓄積する経路情報蓄積手段を備え、前記論理的距離取得手段は、前記経路情報蓄積手段に前記経路情報を蓄積している場合、前記経路情報蓄積手段に蓄積している経路情報から、前記論理的な距離を取得することを特徴とする請求項２８記載のネットワークステータスサーバ。

【請求項３０】　更に、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを蓄積する状態情報蓄積手段を備え、前記収集手段は、前記状態情報蓄積手段に前記状態の情報を蓄積している場合、前記状態情報蓄積手段に蓄積している状態の情報から、前記状態の情報を収集することを特徴とする請求項２７乃至２９のいずれかに記載のネットワークステータスサーバ。

【請求項３１】　前記状態情報蓄積手段は、前記蓄積したサイト内の状態の情報を所定の期間だけ蓄積することを特徴とする請求項３０に記載のネットワークステータスサーバ。

【請求項３２】　前記状態情報蓄積手段は、前記サイト内の状態の情報を所定の近似式で近似して蓄積する請求項３０又は３１のいずれかに記載のネットワークステータスサーバ。

【請求項３３】　前記サイト決定手段は、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間の各々の論理的な距離が最小となるサイトを前記最適なサイトとして決定することを特徴とする請求項２７乃至３２のいずれかに記載のネットワークステータスサーバ。

【請求項３４】　前記第１のサーバ決定手段は、前記最適なサイト内で予め決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする請求項２７乃至３３のいずれかに記載のネットワークステータスサーバ。

【請求項３５】　前記第１のサーバ決定手段は、所定の順序で決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする請求項２７乃至３３のいずれかに記載のネットワークステータスサーバ。

【請求項３６】　前記第１のサーバ決定手段は、前記最適なサイト内の情報配信サーバから任意に情報配信サーバを決定することを特徴とする請求項２７乃至３３のいずれかに記載のネットワークステータスサーバ。

【請求項３７】　前記第１のサーバ決定手段は、前記最適なサイト内の状態の情報に基づいて、最適な情報配信サーバを決定することを特徴とする請求項２７乃至３３のいずれかに記載のネットワークステータスサーバ。

【請求項３８】　前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報を収集する場合、前記収集手段は、前記アクセスしてきたクライアントと前記各サイトとの間の、応答時間、ルータ段数、パケットロス率の内、少なくともどれか一つの情報を収集することを特徴とする請求項２７乃至３７のいずれかに記載のネットワークステータスサーバ。

【請求項３９】　前記サイト内の状態の情報とは、前記サイト内のネットワークの状態の情報と、前記サイト内の情報配信サーバの状態の情報との少なくともどちらか一つであることを特徴とする請求項２７乃至３８のいずれかに記載のネットワークステータスサーバ。

【請求項４０】　前記分散配置された情報配信サーバを有するサイト内のネットワークの状態の情報を収集する場合、前記収集手段は、前記サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともいずれか一つの情報を収集することを特徴とする請求項３９記載のネットワークステータスサーバ。

【請求項４１】　前記分散配置された情報配信サーバの状態の情報を収集する場合、前記収集手段は、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷のうち、少なくともどれか一つの情報を収集することを特徴とする請求項３９又は４０のいずれかに記載のネットワークステータスサーバ。

【請求項４２】　前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を蓄積する経路情報蓄積手段と、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサ

イト内の状態の情報とを蓄積する状態情報蓄積手段とを備え、前記第２のサーバ決定手段は、蓄積された前記経路情報と前記状態の情報と所定の重み係数とに基づいて、下記の式；
・ネットワーク状態値：Ｋ１ｎ＝ＲＴｎ・Ａ＋ＲＮｎ・Ｂ＋ＰＬｎ・Ｃ
・サイト内ネットワーク状態値：Ｋ２ｎ＝ＣＳｎ・Ｄ＋ＰＳｎ・Ｅ＋ＥＳｎ・Ｆ
・サーバ状態値：Ｋ３ｎｍ＝ＣＰＵｎｍ・Ｇ＋ＩＤＬＥｎｍ・Ｈ＋ＬＩＮＫｎｍ
・Ｉ＋ＩＯｎｍ・Ｊ
・最適サイト判定値：Ｋｎ＝Ｋ１ｎ・Ｋ＋Ｋ２ｎ・Ｌ＋ＡＳＬｎ・Ｍ
・最適サーバ判定値：Ｓｎｍ＝Ｋｎ・Ｎ＋Ｋ３ｎｍ・Ｏ
（但し、ｎ：サーバサイト番号、ｍ：サーバ番号、Ａ～Ｏ：重み係数であり、各記号はそれぞれ、ＡＳＬｎ：ＡＳパスによる論理的なネットワーク間距離（ＢＧＰの経路情報を用いる）、ＲＴｎ：応答時間、ＲＮｎ：ルータ段数（ルータホップ数）、ＰＬｎ：パケットロス率、ＣＳｎ：サイト内の混雑度、ＰＳｎ：サイト内のパケット数、ＥＳｎ：パケットエラー、ＣＰＵｎｍ：ＣＰＵ負荷率、ＩＤＬＥｎｍ：ＣＰＵアイドル値、ＬＩＮＫｎｍ：コネクションリンク数、ＩＯｎｍ：ディスク負荷である。）における最適サーバ判定値Ｓｎｍが所定の条件を満たす情報配信サーバを最適な情報配信サーバとして決定することを特徴とする請求項２７記載のネットワークステータスサーバ。

【請求項４３】　分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムであって、前記情報配信サーバは、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせる問い合せ手段と、前記ネットワークステータスサーバからの返答を受け取る返答受信手段と、前記アクセスするクライアントに前記最適な情報配信サーバを指示する指示手段とを備え、前記ネットワークステータスサーバは、前記問い合せ手段からの問い合せを受信する問い合せ受信手段と、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得手段と、前記取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定手段と、前記サイト決定手段により決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定するサーバ決定手段と、前記サーバ決定手段により、決定したサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答する返答送信手段とを備えることを特徴とする情報配信システム。

【請求項４４】　更に、前記情報配信システムはルートサーバを備え、前記ルートサーバはサイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有し、前記論理的距離取得手段が前記論理的な距離を取得する際に、前記論理的な距離を提供することを特徴とする請求項４３記載の情報配信システム。

【請求項４５】　更に、前記情報配信サーバは、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集する手段と、収集した前記ネットワーク状態の情報をネットワークステータスサーバに送信するネットワーク状態送信手段とを備えることを特徴とする請求項４３又は４４のいずれかに記載の情報配信システム。

【請求項４６】　更に、前記情報配信サーバは、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集する手段と、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信するサイト内情報送信手段とを備えることを特徴とする請求項４３乃至４５のいずれかに記載の情報配信システム。

【請求項４７】　分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムであって、前記情報配信サーバは、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせる問い合せ手段と、前記ネットワークステータスサーバからの返答を受け取る返答受信手段と、前記アクセスするクライアントに前記最適な情報配信サーバを指示する指示手段とを備え、前記ネットワークステータスサーバは、前記問い合せ手段からの問い合せを受信する問い合せ受信手段と、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得手段と、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集手段とを有し、前記論理的距離取得手段により取得した論理的な距離と、前記収集手段により収集した前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定するサーバ決定手段と、前記サーバ決定手段により、決定したサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答する返答送信手段とを備えることを特徴とする情報配信システム。

【請求項４８】　更に、前記情報配信システムはルートサーバを備え、前記ルートサーバはサイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有し、前記論理的距離取得手段が前記論理的な距離を取得する際に、前記論理的な距離を提供することを特徴とする請求項４７記載の情報配信システム。

【請求項４９】　更に、前記情報配信サーバは、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集する手段と、収集した前記ネットワーク状態の情報をネットワークステータスサーバに送信するネットワーク状態送信手段とを備えることを特徴とする請求項４７又は４８のいずれかに記載の情報配信システム。

【請求項５０】　更に、前記情報配信サーバは、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集する手段と、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信するサイト内情報送信手段とを備えることを特徴とする請求項４７乃至４９のいずれかに記載の情報配信システム。

【請求項５１】　分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムであって、前記情報配信サーバは、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせる問い合せ手段と、前記ネットワークステータスサーバからの返答を受け取る返答受信手段と、前記アクセスするクライアントに前記最適な情報配信サーバを指示する指示手段とを備え、前記ネットワークステータスサーバは、前記問い合せ手段からの問い合せを受信する問い合せ受信手段と、前記分散配置された情報配信サーバを有する各サイトと前記情報配信サーバにアクセスするクライアントとの間の各々の論理的な距離を取得する論理的距離取得手段と、前記分散配置された情報配信サーバを有する各サイトと前記アクセスするクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集手段とを有し、前記クライアントからのアクセスが最初のアクセスであると判断した場合、前記論理的距離取得手段により取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定手段と、前記サイト決定手段により決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定する第１のサーバ決定手段とを有し、前記クライアントからのアクセスが最初のアクセスでないと判断した場合、前記論理的距離取得手段により取得した論理的な距離と、前記収集手段により収集した前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定する第２のサーバ決定手段とを有し、前記第１のサーバ決定手段もしくは前記第２のサーバ決定手段により、決定したサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答する返答送信手段とを備えることを特徴とする情報配信システム。

【請求項５２】　更に、前記情報配信システムはルートサーバを備え、前記ルートサーバはサイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有し、前記論理的距離取得手段が前記論理的な距離を取得する際に、前記論理的な距離を提供することを特徴とする請求項５１記載の情報配信システム。

【請求項５３】　更に、前記情報配信サーバは、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集する手段と、収集した前記ネットワーク状態の情報をネットワークステータスサーバに送信するネットワーク状態送信手段とを備えることを特徴とする請求項５１又は５２のいずれかに記載の情報配信システム。

【請求項５４】　更に、前記情報配信サーバは、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集する手段と、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信するサイト内情報送信手段とを備えることを特徴とする請求項５１乃至５３のいずれかに記載の情報配信システム。

【請求項５５】　ネットワークステータスサーバを制御するネットワークステータスサーバ制御方法であって、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得ステップと、前記取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定ステップとを備えることを特徴とするネットワークステータスサーバ制御方法。

【請求項５６】　サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得ステップにおいて、前記論理的な距離を取得することを特徴とする請求項５５記載のネットワークステータスサーバ制御方法。

【請求項５７】　前記ネットワークステータスサーバは経路情報蓄積手段を有し、前記ネットワークステータス

サーバ制御方法は、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積する経路情報蓄積ステップを備え、前記論理的距離取得ステップでは、前記経路情報蓄積手段に前記経路情報が蓄積されている場合、前記経路情報蓄積手段に蓄積されている経路情報から、前記論理的な距離を取得することを特徴とする請求項５６記載のネットワークステータスサーバ制御方法。

【請求項５８】　前記サイト決定ステップでは、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間の各々の論理的な距離が最小となるサイトを前記最適なサイトとして決定することを特徴とする請求項５５乃至５７のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項５９】　更に、前記サイト決定ステップにおいて決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定するサーバ決定ステップを備えることを特徴とする請求項５５乃至５８のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項６０】　前記サーバ決定ステップでは、前記最適なサイト内で予め決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする請求項５９記載のネットワークステータスサーバ制御方法。

【請求項６１】　前記サーバ決定ステップでは、所定の順序で決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする請求項５９記載のネットワークステータスサーバ制御方法。

【請求項６２】　前記サーバ決定ステップでは、前記最適なサイト内の情報配信サーバから任意に情報配信サーバを決定することを特徴とする請求項５９記載のネットワークステータスサーバ制御方法。

【請求項６３】　更に、前記分散配置された情報配信サーバを有するサイト内の状態の情報を収集する状態情報収集ステップを有し、前記サーバ決定ステップでは、収集した前記最適なサイト内の状態の情報を条件として、最適な情報配信サーバを決定することを特徴とする請求項５９記載のネットワークステータスサーバ制御方法。

【請求項６４】　前記ネットワークステータスサーバは、状態情報蓄積手段を有し、前記ネットワークステータスサーバ制御方法は、更に、収集した前記サイト内の状態の情報を状態情報蓄積手段に蓄積する状態情報蓄積ステップを備え、前記状態情報蓄積手段に前記最適なサイト内の状態の情報が蓄積されている場合、前記サーバ決定ステップでは、蓄積された前記最適なサイト内の状態の情報を条件として、最適な情報配信サーバを決定することを特徴とする請求項６３記載のネットワークステータスサーバ制御方法。

【請求項６５】　所定の期間を経過後、前記状態情報蓄積手段に蓄積したサイト内の状態の情報を破棄する状態情報破棄ステップを備えることを特徴とする請求項６４に記載のネットワークステータスサーバ制御方法。

【請求項６６】　前記状態情報蓄積ステップでは、前記サイト内の状態の情報を所定の近似式で近似して蓄積することを特徴とする請求項６４又は６５記載のネットワークステータスサーバ制御方法。

【請求項６７】　前記サイト内の状態の情報とは、前記サイト内のネットワークの状態の情報と、前記サイト内の情報配信サーバの状態の情報との少なくともどちらか一つの情報であることを特徴とする請求項６３乃至６６のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項６８】　前記分散配置された情報配信サーバを有する各サイト内のネットワークの状態の情報を収集する場合、前記収集ステップでは、前記各サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともどれか一つを収集することを特徴とする請求項６７記載のネットワークステータスサーバ制御方法。

【請求項６９】　前記分散配置された情報配信サーバの状態の情報を収集する収集ステップでは、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷の内、少なくともいずれか一つを収集することを特徴とする請求項６７又は６８のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項７０】　ネットワークステータスサーバを制御するネットワークステータスサーバ制御方法であって、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集ステップと、前記論理的距離取得ステップにおいて取得した論理的な距離と、前記収集ステップにおいて収集した前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定するサーバ決定ステップとを備えることを特徴とするネットワークステータスサーバ制御方法。

【請求項７１】　サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得ステップにおいて、前記論理的な距離を取得することを特徴とする請求項７０記載のネットワークステータスサーバ制御方法。

【請求項７２】　前記ネットワークステータスサーバは経路情報蓄積手段を有し、前記ネットワークステータスサーバ制御方法は、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積する経路情報蓄積ステップを備え、前記論理的距離取得ステップでは、前記経路情報蓄積手段に前記経路情報を蓄積している場合、前記経路情報蓄積ステップに蓄積している経路情報から、前記論理的な距離を取得することを特徴とする請求項７１記載のネットワークステータスサーバ制御方法。

【請求項７３】　前記ネットワークステータスサーバは状態情報蓄積手段を有し、前記ネットワークステータスサーバ制御方法は、更に、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを状態情報蓄積手段に蓄積する状態情報蓄積ステップを備え、前記収集ステップでは、前記状態情報蓄積手段に前記状態の情報を蓄積している場合、前記状態情報蓄積手段に蓄積している前記状態の情報から、前記状態の情報を収集することを特徴とする請求項７０乃至７２のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項７４】　所定の期間を経過後、前記状態情報蓄積手段に蓄積したサイト内の状態の情報を破棄する状態情報破棄ステップを備えることを特徴とする請求項７３に記載のネットワークステータスサーバ制御方法。

【請求項７５】　前記状態情報蓄積ステップでは、前記サイト内の状態の情報を所定の近似式で近似して蓄積する請求項７３又は７４のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項７６】　前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報を収集する場合、前記収集ステップでは、前記アクセスしてきたクライアントと前記各サイトとの間の、応答時間、ルータ段数、パケットロス率のうち、少なくともどれか一つを収集することを特徴とする請求項７０乃至７５のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項７７】　前記分散配置された情報配信サーバを有するサイト内の状態の情報とは、前記分散配置された情報配信サーバを有するサイト内のネットワークの状態の情報、又は、前記情報配信サーバの状態の情報の少なくともどちらか一つであることを特徴とする請求項７０乃至７６のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項７８】　前記分散配置された情報配信サーバを有する各サイト内のネットワークの状態の情報を収集する場合、前記収集ステップでは、前記各サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともいずれか一つを収集することを特徴とする請求項７７記載のネットワークステータスサーバ制御方法。

【請求項７９】　前記分散配置された情報配信サーバの状態の情報を収集する場合、前記収集ステップでは、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷の内、少なくともいずれか一つを収集することを特徴とする請求項７７又は７８記載のネットワークステータスサーバ制御方法。

【請求項８０】　前記ネットワークステータスサーバは、経路情報蓄積手段と状態情報蓄積手段とを有し、前記ネットワークステータスサーバ制御方法は、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を経路情報蓄積手段に蓄積する経路情報蓄積ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを状態情報蓄積手段に蓄積する状態情報蓄積ステップとを備え、前記サーバ決定ステップでは、蓄積された前記経路情報と前記状態の情報と所定の重み係数とに基づいて、下記の式；

・ネットワーク状態値：Ｋ１ｎ＝ＲＴｎ・Ａ＋ＲＮｎ・Ｂ＋ＰＬｎ・Ｃ

・サイト内ネットワーク状態値：Ｋ２ｎ＝ＣＳｎ・Ｄ＋ＰＳｎ・Ｅ＋ＥＳｎ・Ｆ

・サーバ状態値：Ｋ３ｎｍ＝ＣＰＵｎｍ・Ｇ＋ＩＤＬＥｎｍ・Ｈ＋ＬＩＮＫｎｍ

・Ｉ＋ＩＯｎｍ・Ｊ

・最適サイト判定値：Ｋｎ＝Ｋ１ｎ・Ｋ＋Ｋ２ｎ・Ｌ＋ＡＳＬｎ・Ｍ

・最適サーバ判定値：Ｓｎｍ＝Ｋｎ・Ｎ＋Ｋ３ｎｍ・Ｏ

（但し、ｎ：サーバサイト番号、ｍ：サーバ番号、Ａ～Ｏ：重み係数であり、各記号はそれぞれ、ＡＳＬｎ：ＡＳパスによる論理的なネットワーク間距離（ＢＧＰの経路情報を用いる）、ＲＴｎ：応答時間、ＲＮｎ：ルータ段数（ルータホップ数）、ＰＬｎ：パケットロス率、ＣＳｎ：サイト内の混雑度、ＰＳｎ：サイト内のパケット数、ＥＳｎ：パケットエラー、ＣＰＵｎｍ：ＣＰＵ負荷率、ＩＤＬＥｎｍ：ＣＰＵアイドル値、ＬＩＮＫｎｍ：コネクションリンク数、ＩＯｎｍ：ディスク負荷である。）における最適サーバ判定値Ｓｎｍが所定の条件を満たす情報配信サーバを最適な情報配信サーバとして決定することを特徴とする請求項７０記載のネットワークステータスサーバ制御方法。

【請求項８１】　分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイ

トと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集ステップとを有し、前記クライアントからのアクセスが最初のアクセスであると判断した場合、前記論理的距離取得ステップにおいて取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定ステップと、前記サイト決定ステップにおいて決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定する第１のサーバ決定ステップとを備え、前記クライアントからのアクセスが最初のアクセスでないと判断した場合、前記論理的距離取得ステップにおいて取得した論理的な距離と、前記収集ステップにおいて収集した前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定する第２のサーバ決定ステップとを備えることを特徴とするネットワークステータスサーバ制御方法。

【請求項８２】　サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得ステップにおいて、前記論理的な距離を取得することを特徴とする請求項８１記載のネットワークステータスサーバ制御方法。

【請求項８３】　前記ネットワークステータスサーバは経路情報蓄積手段を有し、前記ネットワークステータスサーバ制御方法は、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積する経路情報蓄積ステップを備え、前記論理的距離取得ステップでは、前記経路情報蓄積手段に前記経路情報を蓄積している場合、前記経路情報蓄積手段に蓄積している経路情報から、前記論理的な距離を取得することを特徴とする請求項８２記載のネットワークステータスサーバ制御方法。

【請求項８４】　前記ネットワークステータスサーバは状態情報蓄積手段を有し、前記ネットワークステータスサーバ制御方法は、更に、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを前記状態情報蓄積手段に蓄積する状態情報蓄積ステップを備え、前記収集ステップでは、前記状態情報蓄積手段に前記状態の情報を蓄積している場合、前記状態情報蓄積手段に蓄積している前記状態の情報から、前記状態の情報を収集することを特徴とする請求項８１乃至８３のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項８５】　所定の期間を経過後、前記状態情報蓄積手段に蓄積したサイト内の状態の情報を破棄する状態情報破棄ステップを備えることを特徴とする請求項８４に記載のネットワークステータスサーバ制御方法。

【請求項８６】　前記状態情報蓄積ステップでは、前記サイト内の状態の情報を所定の近似式で近似して蓄積する請求項８４又は８５のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項８７】　前記サイト決定ステップでは、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間の各々の論理的な距離が最小となるサイトを前記最適なサイトとして決定することを特徴とする請求項８１乃至８６のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項８８】　前記第１のサーバ決定ステップでは、前記最適なサイト内で予め決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする請求項８１乃至８７のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項８９】　前記第１のサーバ決定ステップでは、所定の順序で決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする請求項８１乃至８７のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項９０】　前記第１のサーバ決定ステップでは、前記最適なサイト内の情報配信サーバから任意に情報配信サーバを決定することを特徴とする請求項８１乃至８７のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項９１】　前記第１のサーバ決定ステップでは、前記最適なサイト内の状態の情報に基づいて、最適な情報配信サーバを決定することを特徴とする請求項８１乃至８７のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項９２】　前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報を収集する場合、前記収集ステップでは、前記アクセスしてきたクライアントと前記各サイトとの間の、応答時間、ルータ段数、パケットロス率の内、少なくともどれか一つの情報を収集することを特徴とする請求項８１乃至９１のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項９３】　前記サイト内の状態の情報とは、前記サイト内のネットワークの状態の情報と、前記サイト内の情報配信サーバの状態の情報との少なくともどちらか一つであることを特徴とする請求項８１乃至９２のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項９４】　前記分散配置された情報配信サーバを

有するサイト内のネットワークの状態の情報を収集する場合、前記収集ステップでは、前記サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともいずれか一つの情報を収集することを特徴とする請求項９３記載のネットワークステータスサーバ制御方法。

【請求項９５】　前記分散配置された情報配信サーバの状態の情報を収集する場合、前記収集ステップでは、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷のうち、少なくともどれか一つの情報を収集することを特徴とする請求項９３又は９４のいずれかに記載のネットワークステータスサーバ制御方法。

【請求項９６】　前記ネットワークステータスサーバは、経路情報蓄積手段と状態情報蓄積手段とを有し、前記ネットワークステータスサーバ制御方法は、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積する経路情報蓄積ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを前記状態情報蓄積手段に蓄積する状態情報蓄積ステップとを備え、前記第２のサーバ決定ステップでは、蓄積された前記経路情報と前記状態の情報と所定の重み係数とに基づいて、下記の式；

・ネットワーク状態値：Ｋ１ｎ＝ＲＴｎ・Ａ＋ＲＮｎ・Ｂ＋ＰＬｎ・Ｃ

・サイト内ネットワーク状態値：Ｋ２ｎ＝ＣＳｎ・Ｄ＋ＰＳｎ・Ｅ＋ＥＳｎ・Ｆ

・サーバ状態値：Ｋ３ｎｍ＝ＣＰＵｎｍ・Ｇ＋ＩＤＬＥｎｍ・Ｈ＋ＬＩＮＫｎｍ

・Ｉ＋ＩＯｎｍ・Ｊ

・最適サイト判定値：Ｋｎ＝Ｋ１ｎ・Ｋ＋Ｋ２ｎ・Ｌ＋ＡＳＬｎ・Ｍ

・最適サーバ判定値：Ｓｎｍ＝Ｋｎ・Ｎ＋Ｋ３ｎｍ・Ｏ

（但し、ｎ：サーバサイト番号、ｍ：サーバ番号、Ａ～Ｏ：重み係数であり、各記号はそれぞれ、ＡＳＬｎ：ＡＳパスによる論理的なネットワーク間距離（ＢＧＰの経路情報を用いる）、ＲＴｎ：応答時間、ＲＮｎ：ルータ段数（ルータホップ数）、ＰＬｎ：パケットロス率、ＣＳｎ：サイト内の混雑度、ＰＳｎ：サイト内のパケット数、ＥＳｎ：パケットエラー、ＣＰＵｎｍ：ＣＰＵ負荷率、ＩＤＬＥｎｍ：ＣＰＵアイドル値、ＬＩＮＫｎｍ：コネクションリンク数、ＩＯｎｍ：ディスク負荷である。）における最適サーバ判定値Ｓｎｍが所定の条件を満たす情報配信サーバを最適な情報配信サーバとして決定することを特徴とする請求項８１記載のネットワークステータスサーバ制御方法。

【請求項９７】　分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムを制御する情報配信システム制御方法であって、前記情報配信サーバにおいて、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせる問い合せステップと、前記ネットワークステータスサーバからの返答を受け取る返答受信ステップと、前記アクセスするクライアントに前記最適な情報配信サーバを指示する指示ステップとを備え、前記ネットワークステータスサーバにおいて、前記問い合せステップからの問い合せを受信する問い合せ受信ステップと、分散配置された情報配信サーバを有する各サイトとアクセスするクライアントとの間の各々の論理的な距離を取得する論理的距離取得ステップと、前記取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定ステップと、前記サイト決定ステップにより決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定するサーバ決定ステップと、前記サーバ決定ステップにおいて決定したサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答する返答送信ステップとを備えることを特徴とする情報配信システム制御方法。

【請求項９８】　更に、前記情報配信システムはルートサーバを有し、前記ルートサーバにおいて、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める論理的距離算出ステップを備え、前記論理的距離取得ステップでは、前記論理的距離算出ステップで算出した前記論理的な距離を取得することを特徴とする請求項９７記載の情報配信システム制御方法。

【請求項９９】　更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集するステップと、収集した前記ネットワーク状態の情報をネットワークステータスサーバに送信するネットワーク状態送信ステップとを備えることを特徴とする請求項９７又は９８のいずれかに記載の情報配信システム制御方法。

【請求項１００】　更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集するステップと、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信するサイト内情報送信ステップとを備えることを特徴とする請求項９７乃至９９のいずれかに記載の情報配信システム制御方法。

【請求項１０１】　分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムを制御する情報配信システム制御方法であって、前記情報配信サーバにおいて、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサ

ーバに問い合わせる問い合せステップと、前記ネットワークステータスサーバからの返答を受け取る返答受信ステップと、前記アクセスするクライアントに前記最適な情報配信サーバを指示する指示ステップとを備え、前記ネットワークステータスサーバにおいて、前記問い合せステップからの問い合せを受信する問い合せ受信ステップと、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集ステップと、前記論理的距離取得ステップにおいて取得した論理的な距離と、前記収集ステップにおいて収集した、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定するサーバ決定ステップと、前記サーバ決定ステップにおいて決定したサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答する返答送信ステップとを備えることを特徴とする情報配信システム制御方法。

【請求項１０２】　更に、前記情報配信システムはルートサーバを有し、前記ルートサーバにおいて、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める論理的距離算出ステップを備え、前記論理的距離取得ステップでは、前記論理的距離算出ステップで算出した前記論理的な距離を取得することを特徴とする請求項１０１記載の情報配信システム制御方法。

【請求項１０３】　更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集するステップと、収集した前記ネットワーク状態の情報をネットワークステータスサーバに送信するネットワーク状態送信ステップとを備えることを特徴とする請求項１０１又は１０２のいずれかに記載の情報配信システム制御方法。

【請求項１０４】　更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集するステップと、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信するサイト内情報送信ステップとを備えることを特徴とする請求項１０１乃至１０３のいずれかに記載の情報配信システム。

【請求項１０５】　分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムを制御する情報配信システム制御方法であって、前記情報配信サーバにおいて、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせる問い合せステップと、前記ネットワークステータスサーバからの返答を受け取る返答受信ステップと、前記アクセスするクライアントに前記最適な情報配信サーバを指示する指示ステップとを備え、前記ネットワークステータスサーバにおいて、前記問い合せステップからの問い合せを受信する問い合せ受信ステップと、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集ステップと、前記クライアントからのアクセスが最初のアクセスであると判断した場合、前記論理的距離取得ステップにおいて取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定ステップと、前記サイト決定ステップにおいて決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定する第１のサーバ決定ステップと、前記クライアントからのアクセスが最初のアクセスでないと判断した場合、前記論理的距離取得ステップにおいて取得した論理的な距離と、前記収集ステップにおいて収集した、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定する第２のサーバ決定ステップと、前記第１のサーバ決定ステップもしくは前記第２のサーバ決定ステップにおいて決定したサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答する返答送信ステップとを備えることを特徴とする情報配信システム制御方法。

【請求項１０６】　更に、前記情報配信システムはルートサーバを備え、前記ルートサーバにおいて、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める論理的距離算出ステップを備え、前記論理的距離取得ステップでは、前記論理的距離算出ステップで算出した前記論理的な距離を取得することを特徴とする請求項１０５記載の情報配信システム制御方法。

【請求項１０７】　更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトと前記アクセスす

るクライアントの間のネットワーク状態の情報を収集するステップと、収集した前記ネットワーク状態の情報をネットワークステータスサーバに送信するネットワーク状態送信ステップとを備えることを特徴とする請求項１０５又は１０６のいずれかに記載の情報配信システム制御方法。

【請求項１０８】　更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集するステップと、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信するサイト内情報送信ステップとを備えることを特徴とする請求項１０５乃至１０７のいずれかに記載の情報配信システム。

【請求項１０９】　ネットワークステータスサーバを制御するコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体であって、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得させる論理的距離取得ステップと、前記取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定させるサイト決定ステップとを備えることを特徴とするコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１１０】　サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得ステップにおいて、前記論理的な距離を取得させることを特徴とする請求項１０９記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１１１】　前記ネットワークステータスサーバは経路情報蓄積手段を有し、前記ネットワークステータスサーバ制御プログラムは、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積させる経路情報蓄積ステップを備え、前記論理的距離取得ステップでは、前記経路情報蓄積手段に前記経路情報が蓄積されている場合、前記経路情報蓄積手段に蓄積されている経路情報から、前記論理的な距離を取得させることを特徴とする請求項１１０記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１１２】　前記サイト決定ステップでは、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間の各々の論理的な距離が最小となるサイトを前記最適なサイトとして決定させることを特徴とする請求項１０９乃至１１１のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１１３】　更に、前記サイト決定ステップにおいて決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定させるサーバ決定ステップを備えることを特徴とする請求項１０９乃至１１２のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１１４】　前記サーバ決定ステップでは、前記最適なサイト内で予め決められた情報配信サーバを、最適な情報配信サーバとして決定させることを特徴とする請求項１１３記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１１５】　前記サーバ決定ステップでは、所定の順序で決められた情報配信サーバを、最適な情報配信サーバとして決定させることを特徴とする請求項１１３記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１１６】　前記サーバ決定ステップでは、前記最適なサイト内の情報配信サーバから任意に情報配信サーバを決定させることを特徴とする請求項１１３記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１１７】　更に、前記分散配置された情報配信サーバを有するサイト内の状態の情報を収集させる状態情報収集ステップを有し、前記サーバ決定ステップでは、収集した前記最適なサイト内の状態の情報を条件として、最適な情報配信サーバを決定させることを特徴とする請求項１１３記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１１８】　前記ネットワークステータスサーバは、状態情報蓄積手段を有し、前記ネットワークステータスサーバ制御プログラムは、更に、収集した前記サイト内の状態の情報を状態情報蓄積手段に蓄積させる状態情報蓄積ステップを備え、前記状態情報蓄積手段に前記最適なサイト内の状態の情報が蓄積されている場合、前記サーバ決定ステップでは、蓄積された前記最適なサイト内の状態の情報を条件として、最適な情報配信サーバを決定させることを特徴とする請求項１１７記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１１９】　所定の期間を経過後、前記状態情報蓄積手段に蓄積したサイト内の状態の情報を破棄させる状態情報破棄ステップを備えることを特徴とする請求項１１８に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１２０】　前記状態情報蓄積ステップでは、前

記サイト内の状態の情報を所定の近似式で近似して蓄積させることを特徴とする請求項１１８又は１１９記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１２１】　前記サイト内の状態の情報とは、前記サイト内のネットワークの状態の情報と、前記サイト内の情報配信サーバの状態の情報との少なくともどちらか一つの情報であることを特徴とする請求項１１７乃至１２０のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１２２】　前記分散配置された情報配信サーバを有する各サイト内のネットワークの状態の情報を収集させる場合、前記収集ステップでは、前記各サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともどれか一つを収集させることを特徴とする請求項１２１記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１２３】　前記分散配置された情報配信サーバの状態の情報を収集させる収集ステップでは、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷の内、少なくともいずれか一つを収集させることを特徴とする請求項１２１又は１２２のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１２４】　ネットワークステータスサーバを制御するコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体であって、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得させる論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集させる収集ステップと、前記論理的距離取得ステップにおいて取得させた論理的な距離と、前記収集ステップにおいて収集させた、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定させるサーバ決定ステップとを備えることを特徴とするコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１２５】　サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得ステップにおいて、前記論理的な距離を取得させることを特徴とする請求項１２４記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１２６】　前記ネットワークステータスサーバは経路情報蓄積手段を有し、前記ネットワークステータスサーバ制御プログラムは、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積させる経路情報蓄積ステップを備え、前記論理的距離取得ステップでは、前記経路情報蓄積手段に前記経路情報を蓄積している場合、前記経路情報蓄積ステップに蓄積している経路情報から、前記論理的な距離を取得させることを特徴とする請求項１２５記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１２７】　前記ネットワークステータスサーバは状態情報蓄積手段を有し、前記ネットワークステータスサーバ制御プログラムは、更に、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを状態情報蓄積手段に蓄積させる状態情報蓄積ステップを備え、前記収集ステップでは、前記状態情報蓄積手段に前記状態の情報を蓄積している場合、前記状態情報蓄積手段に蓄積している前記状態の情報から、前記状態の情報を収集させることを特徴とする請求項１２４乃至１２６のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１２８】　所定の期間を経過後、前記状態情報蓄積手段に蓄積したサイト内の状態の情報を破棄させる状態情報破棄ステップを備えることを特徴とする請求項１２７に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１２９】　前記状態情報蓄積ステップでは、前記サイト内の状態の情報を所定の近似式で近似して蓄積させる請求項１２７又は１２８のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１３０】　前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報を収集させる場合、前記収集ステップでは、前記アクセスしてきたクライアントと前記各サイトとの間の、応答時間、ルータ段数、パケットロス率のうち、少なくともどれか一つを収集させることを特徴とする請求項１２４乃至１２９のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１３１】　前記分散配置された情報配信サーバを有するサイト内の状態の情報とは、前記分散配置された情報配信サーバを有するサイト内のネットワークの状態の情報、又は、前記情報配信サーバの状態の情報の少なくともどちらか一つであることを特徴とする請求項１２４乃至１３０のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１３２】　前記分散配置された情報配信サーバを有する各サイト内のネットワークの状態の情報を収集させる場合、前記収集ステップでは、前記各サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともいずれか一つを収集させることを特徴とする請求項１３１記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１３３】　前記分散配置された情報配信サーバの状態の情報を収集させる場合、前記収集ステップでは、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷の内、少なくともいずれか一つを収集させることを特徴とする請求項１３１又は１３２記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１３４】　前記ネットワークステータスサーバは、経路情報蓄積手段と状態情報蓄積手段とを有し、前記ネットワークステータスサーバ制御プログラムは、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を経路情報蓄積手段に蓄積させる経路情報蓄積ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを状態情報蓄積手段に蓄積させる状態情報蓄積ステップとを備え、前記サーバ決定ステップでは、蓄積された前記経路情報と前記状態の情報と所定の重み係数とに基づいて、下記の式；

・ネットワーク状態値：Ｋ１ｎ＝ＲＴｎ・Ａ＋ＲＮｎ・Ｂ＋ＰＬｎ・Ｃ

・サイト内ネットワーク状態値：Ｋ２ｎ＝ＣＳｎ・Ｄ＋ＰＳｎ・Ｅ＋ＥＳｎ・Ｆ

・サーバ状態値：Ｋ３ｎｍ＝ＣＰＵｎｍ・Ｇ＋ＩＤＬＥｎｍ・Ｈ＋ＬＩＮＫｎｍ

・Ｉ＋ＩＯｎｍ・Ｊ

・最適サイト判定値：Ｋｎ＝Ｋ１ｎ・Ｋ＋Ｋ２ｎ・Ｌ＋ＡＳＬｎ・Ｍ

・最適サーバ判定値：Ｓｎｍ＝Ｋｎ・Ｎ＋Ｋ３ｎｍ・Ｏ

（但し、ｎ：サーバサイト番号、ｍ：サーバ番号、Ａ～Ｏ：重み係数であり、各記号はそれぞれ、ＡＳＬｎ：ＡＳパスによる論理的なネットワーク間距離（ＢＧＰの経路情報を用いる）、ＲＴｎ：応答時間、ＲＮｎ：ルータ段数（ルータホップ数）、ＰＬｎ：パケットロス率、ＣＳｎ：サイト内の混雑度、ＰＳｎ：サイト内のパケット数、ＥＳｎ：パケットエラー、ＣＰＵｎｍ：ＣＰＵ負荷率、ＩＤＬＥｎｍ：ＣＰＵアイドル値、ＬＩＮＫｎｍ：コネクションリンク数、ＩＯｎｍ：ディスク負荷である。）における最適サーバ判定値Ｓｎｍが所定の条件を満たす情報配信サーバを最適な情報配信サーバとして決定させることを特徴とする請求項１２４記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１３５】　分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得させる論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集させる収集ステップとを有し、前記クライアントからのアクセスが最初のアクセスであると判断した場合、前記論理的距離取得ステップにおいて取得させた論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定させるサイト決定ステップと、前記サイト決定ステップにおいて決定させた最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定させる第１のサーバ決定ステップとを備え、前記クライアントからのアクセスが最初のアクセスでないと判断した場合、前記論理的距離取得ステップにおいて取得させた論理的な距離と、前記収集ステップにおいて収集させた前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定させる第２のサーバ決定ステップとを備えることを特徴とするコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１３６】　サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得ステップにおいて、前記論理的な距離を取得させることを特徴とする請求項１３５記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１３７】　前記ネットワークステータスサーバは経路情報蓄積手段を有し、前記ネットワークステータスサーバ制御プログラムは、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に

蓄積させる経路情報蓄積ステップを備え、前記論理的距離取得ステップでは、前記経路情報蓄積手段に前記経路情報を蓄積している場合、前記経路情報蓄積手段に蓄積している経路情報から、前記論理的な距離を取得させることを特徴とする請求項１３６記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１３８】　前記ネットワークステータスサーバは状態情報蓄積手段を有し、前記ネットワークステータスサーバ制御プログラムは、更に、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを前記状態情報蓄積手段に蓄積させる状態情報蓄積ステップを備え、前記収集ステップでは、前記状態情報蓄積手段に前記状態の情報を蓄積している場合、前記状態情報蓄積手段に蓄積している前記状態の情報から、前記状態の情報を収集させることを特徴とする請求項１３５乃至１３７のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１３９】　所定の期間を経過後、前記状態情報蓄積手段に蓄積したサイト内の状態の情報を破棄させる状態情報破棄ステップを備えることを特徴とする請求項１３８に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１４０】　前記状態情報蓄積ステップでは、前記サイト内の状態の情報を所定の近似式で近似して蓄積させる請求項１３８又は１３９のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１４１】　前記サイト決定ステップでは、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間の各々の論理的な距離が最小となるサイトを前記最適なサイトとして決定させることを特徴とする請求項１３５乃至１４０のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１４２】　前記第１のサーバ決定ステップでは、前記最適なサイト内で予め決められた情報配信サーバを、最適な情報配信サーバとして決定させることを特徴とする請求項１３５乃至１４１のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１４３】　前記第１のサーバ決定ステップでは、所定の順序で決められた情報配信サーバを、最適な情報配信サーバとして決定させることを特徴とする請求項１３５乃至１４１のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１４４】　前記第１のサーバ決定ステップでは、前記最適なサイト内の情報配信サーバから任意に情報配信サーバを決定させることを特徴とする請求項１３５乃至１４１のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１４５】　前記第１のサーバ決定ステップでは、前記最適なサイト内の状態の情報に基づいて、最適な情報配信サーバを決定させることを特徴とする請求項１３５乃至１４１のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１４６】　前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報を収集させる場合、前記収集ステップでは、前記アクセスしてきたクライアントと前記各サイトとの間の、応答時間、ルータ段数、パケットロス率の内、少なくともどれか一つの情報を収集させることを特徴とする請求項１３５乃至１４５のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１４７】　前記サイト内の状態の情報とは、前記サイト内のネットワークの状態の情報と、前記サイト内の情報配信サーバの状態の情報との少なくともどちらか一つであることを特徴とする請求項１３５乃至１４６のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１４８】　前記分散配置された情報配信サーバを有するサイト内のネットワークの状態の情報を収集させる場合、前記収集ステップでは、前記サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともいずれか一つの情報を収集させることを特徴とする請求項１４７記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１４９】　前記分散配置された情報配信サーバの状態の情報を収集させる場合、前記収集ステップでは、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷のうち、少なくともどれか一つの情報を収集させることを特徴とする請求項１４７又は１４８のいずれかに記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１５０】　前記ネットワークステータスサーバは、経路情報蓄積手段と状態情報蓄積手段とを有し、前記ネットワークステータスサーバ制御プログラムは、前記分散配置された情報配信サーバを有する各サイトとア

クセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積させる経路情報蓄積ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを前記状態情報蓄積手段に蓄積させる状態情報蓄積ステップとを備え、前記第２のサーバ決定ステップでは、蓄積された前記経路情報と前記状態の情報と所定の重み係数とに基づいて、下記の式；

・ネットワーク状態値：Ｋ１ｎ＝ＲＴｎ・Ａ＋ＲＮｎ・Ｂ＋ＰＬｎ・Ｃ

・サイト内ネットワーク状態値：Ｋ２ｎ＝ＣＳｎ・Ｄ＋ＰＳｎ・Ｅ＋ＥＳｎ・Ｆ

・サーバ状態値：Ｋ３ｎｍ＝ＣＰＵｎｍ・Ｇ＋ＩＤＬＥｎｍ・Ｈ＋ＬＩＮＫｎｍ

・Ｉ＋ＩＯｎｍ・Ｊ

・最適サイト判定値：Ｋｎ＝Ｋ１ｎ・Ｋ＋Ｋ２ｎ・Ｌ＋ＡＳＬｎ・Ｍ

・最適サーバ判定値：Ｓｎｍ＝Ｋｎ・Ｎ＋Ｋ３ｎｍ・Ｏ

（但し、ｎ：サーバサイト番号、ｍ：サーバ番号、Ａ～Ｏ：重み係数であり、各記号はそれぞれ、ＡＳＬｎ：ＡＳパスによる論理的なネットワーク間距離（ＢＧＰの経路情報を用いる）、ＲＴｎ：応答時間、ＲＮｎ：ルータ段数（ルータホップ数）、ＰＬｎ：パケットロス率、ＣＳｎ：サイト内の混雑度、ＰＳｎ：サイト内のパケット数、ＥＳｎ：パケットエラー、ＣＰＵｎｍ：ＣＰＵ負荷率、ＩＤＬＥｎｍ：ＣＰＵアイドル値、ＬＩＮＫｎｍ：コネクションリンク数、ＩＯｎｍ：ディスク負荷である。）における最適サーバ判定値Ｓｎｍが所定の条件を満たす情報配信サーバを最適な情報配信サーバとして決定させることを特徴とする請求項１３５記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体。

【請求項１５１】　分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムを制御するコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体であって、前記情報配信サーバにおいて、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせさせる問い合せステップと、前記ネットワークステータスサーバからの返答を受け取らせる返答受信ステップと、前記アクセスするクライアントに前記最適な情報配信サーバを指示させる指示ステップとを備え、前記ネットワークステータスサーバにおいて、前記問い合せステップからの問い合せを受信させる問い合せ受信ステップと、分散配置された情報配信サーバを有する各サイトとアクセスするクライアントとの間の各々の論理的な距離を取得させる論理的距離取得ステップと、前記取得させた論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定させるサイト決定ステップと、前記サイト決定ステップにより決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定させるサーバ決定ステップと、前記サーバ決定ステップにおいて決定されたサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答させる返答送信ステップとを備えることを特徴とするコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体。

【請求項１５２】　更に、前記情報配信システムはルートサーバを有し、前記ルートサーバにおいて、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求めさせる論理的距離算出ステップを備え、前記論理的距離取得ステップでは、前記論理的距離算出ステップで算出させた前記論理的な距離を取得させることを特徴とする請求項１５１記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体。

【請求項１５３】　更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集させるステップと、収集させた前記ネットワーク状態の情報をネットワークステータスサーバに送信させるネットワーク状態送信ステップとを備えることを特徴とする請求項１５１又は１５２のいずれかに記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体。

【請求項１５４】　更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集させるステップと、収集させたサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信させるサイト内情報送信ステップとを備えることを特徴とする請求項１５１乃至１５３のいずれかに記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体。

【請求項１５５】　分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムを制御するコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体であって、前記情報配信サーバにおいて、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせさせる問い合せステップと、前記ネットワークステータスサーバからの返答を受け取らせる返答受信ステップと、前記アクセスするクライアントに前記最適な情報配信サーバを指示させる指示ステップとを

備え、前記ネットワークステータスサーバにおいて、前記問い合せステップからの問い合せを受信させる問い合せ受信ステップと、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得させる論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集させる収集ステップと、前記論理的距離取得ステップにおいて取得させた論理的な距離と、前記収集ステップにおいて収集させた、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定させるサーバ決定ステップと、前記サーバ決定ステップにおいて決定させたサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答させる返答送信ステップとを備えることを特徴とするコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体。

【請求項１５６】　更に、前記情報配信システムはルートサーバを有し、前記ルートサーバにおいて、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求めさせる論理的距離算出ステップを備え、前記論理的距離取得ステップでは、前記論理的距離算出ステップで算出させた前記論理的な距離を取得させることを特徴とする請求項１５５記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体。

【請求項１５７】　更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集させるステップと、収集させた前記ネットワーク状態の情報をネットワークステータスサーバに送信させるネットワーク状態送信ステップとを備えることを特徴とする請求項１５５又は１５６のいずれかに記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体。

【請求項１５８】　更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集させるステップと、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信させるサイト内情報送信ステップとを備えることを特徴とする請求項１５５乃至１５７のいずれかに記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体。

【請求項１５９】　分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムを制御するコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体であって、前記情報配信サーバにおいて、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせさせる問い合せステップと、前記ネットワークステータスサーバからの返答を受け取らせる返答受信ステップと、前記アクセスするクライアントに前記最適な情報配信サーバを指示させる指示ステップとを備え、前記ネットワークステータスサーバにおいて、前記問い合せステップからの問い合せを受信させる問い合せ受信ステップと、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得させる論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集させる収集ステップと、前記クライアントからのアクセスが最初のアクセスであると判断した場合、前記論理的距離取得ステップにおいて取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定させるサイト決定ステップと、前記サイト決定ステップにおいて決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定させる第１のサーバ決定ステップと、前記クライアントからのアクセスが最初のアクセスでないと判断した場合、前記論理的距離取得ステップにおいて取得させた論理的な距離と、前記収集ステップにおいて収集させた、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定させる第２のサーバ決定ステップと、前記第１のサーバ決定ステップもしくは前記第２のサーバ決定ステップにおいて決定させたサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答させる返答送信ステップとを備えることを特徴とするコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体。

【請求項１６０】　更に、前記情報配信システムはルートサーバを備え、前記ルートサーバにおいて、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める論理的距離算出ステップを備え、前記論理的距離取得ステップでは、前記論理的距離算出ステップで求めた前記論理的な距離を取得させることを特徴とする請求項１５９記載のコンピュータ読み取り可能

な情報配信システム制御プログラムを格納した記憶媒体。
【請求項１６１】　更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集させるステップと、収集させた前記ネットワーク状態の情報をネットワークステータスサーバに送信させるネットワーク状態送信ステップとを備えることを特徴とする請求項１５９又は１６０のいずれかに記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体。
【請求項１６２】　更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集させるステップと、収集させたサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信させるサイト内情報送信ステップとを備えることを特徴とする請求項１５９乃至１６１のいずれかに記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、ネットワーク上において大容量データを配信する際におけるネットワークステータスサーバ、情報配信システム、とその制御方法及びその制御プログラムを格納した記憶媒体に関するものである。
【０００２】特に、複数に分散配置された情報配信サーバを制御する際におけるネットワークステータスサーバ、情報配信システム、とその制御方法及びその制御プログラムを格納した記憶媒体に関するものである。
【０００３】
【従来の技術】インターネットを用いた情報配信手法の一つとして、ＷｏｒｌｄＷｉｄｅＷｅｂ（以降Ｗｅｂと称する）が一般的に用いられるようになってきた。この手法は、一般的なコンピュータ（情報処理装置）上でＷｅｂサーバソフトを稼働させ、ＨＴＭＬ言語で記述されたコンテンツ（文字、図、表など）を情報配信するものである。サーバにアクセスするクライアントとなるコンピュータ（以降アクセスクライアントと称する）は、そのサーバとネットワークで接続されていれば良く、汎用の閲覧ソフト（ブラウザー）で自由に閲覧する事が可能である。最近では、全世界をアクセス可能対象としたＷｅｂサーバサイトも出現しその重要性は一段と高まってきている。しかし、一方でこのＷｅｂサーバが、全世界から集中してアクセスをされた場合には、接続ネットワーク、及び、Ｗｅｂサーバに大きな負荷がかかる事になる。そこで、その対策として、現在、いくつかの手法が提案されている。この手法について簡単に図６，図７をもとに説明する。
【０００４】まず、Ｗｅｂサーバに集中するアクセスを分散させる手法として、図６のようなバーチャルホストを利用する方式がある。これは、前述の負荷を複数のサーバで分散させる事を目的としている。この例では、アクセスクライアントが、www.abcde.jpなるＵＲＬ（アクセスアドレス）にアクセスした場合を考えている。www.abcde.jpなるアクセスがあると、まずバーチャルホストにアクセスされる。バーチャルホストでは、そのアクセスを配下の各Ｗｅｂサーバの負荷状況から、適切にアクセスの分散を図る。実際には、www1,www2,www3のいづれかでクライアントからのアクセスに対して所定の情報を配信するのだが、バーチャルホストで一度アクセスを受ける事によって適切なアクセスの分散を図っている。また、他に、ＴＣＰコネクションホップによる転送方式も提案されていいる。図７を用いて説明する。クライアントからアクセスがあると、まず、ｗｗｗ１にアクセスされる。ｗｗｗ１では、スケジューラーが動作しており、このスケジューラーで最適なＷｅｂサーバｗｗｗ３を選択し、クライアントからのアクセスに応答する。つまり、ｗｗｗ１が常にアクセス要求に応え、最適Ｗｅｂサーバ（ｗｗｗ１、ｗｗｗ２、ｗｗｗ３）が応答する。この方式では、スケジュール機能においてアクセスの分散を図っている。
【０００５】
【発明が解決しようとする課題】しかしながら、上記従来のアクセス分散方式では、以下のような問題点が存在する。
１）図６で述べたバーチャルホストを利用した場合には、専用のバーチャルホストの役目をする装置が別途必要であるという問題点があった。
２）また、専用のバーチャルホストですべてのアクセスを一時受け付けるために、多くのアクセスがあると、このバーチャルホストの処理能力が問題となった。つまり、実質バーチャルホストの処理能力がこのＷｅｂサーバサイト（Ｗｅｂサーバの稼動しているサイト）の処理能力になってしまうという事である。
３）また、バーチャルホスト方式は、あくまでそのバーチャルホストの存在するサーバサイト内（自サーバサイト内）のアクセスについて受付け、自サーバサイト内のＷｅｂサーバのアクセスを平準化するため、分散配置されたＷｅｂサーバを有する各Ｗｅｂサーバサイトには、各々バーチャルホストが必要になる。そして、各々のバーチャルホストに所定の異なるＵＲＬを設定する必要があった。そのため、アクセスするクライアントが、分散配置されているＷｅｂサーバサイトを自ら選択しなければならないという問題点があった。
４）さらにアクセスするクライアントが、Ｗｅｂサーバサイトを選択するために、各Ｗｅｂサーバサイトへのアクセスを平準化する事ができないという問題もあった。

【０００６】次に図７で述べたＴＣＰコネクションホップによる転送方式については、バーチャルホスト方式と違い専用のアクセスを一時受け付けるような機構が別途必要でないという利点がある。
５）しかし、常にスケジューラが常駐しているｗｅｂサーバにアクセスが集中するため、万一、スケジューラが常駐しているｗｅｂサーバに障害が発生した場合には、このサーバサイト全体が機能できなくなるという問題点があった。
６）また、この方式においても、分散配置されたｗｅｂサーバサイトへのアクセスを最適に制御する事ができない。それは、あくまで、スケジューラが管理しているｗｅｂサーバサイト（自ネットワーク内）内でのアクセスの平準化を目的としている方式であるため、前述のバーチャルサイト方式と同様に、各々のｗｅｂサーバサイトに所定の異なるＵＲＬを設定する必要があった。そのため、アクセスするクライアントが、分散配置されているｗｅｂサーバサイトを自ら選択するという問題点があった。
７）さらに前述のバーチャルサイト方式と同様に、アクセスするクライアントがｗｅｂサーバサイトを選択するために、各ｗｅｂサーバサイトへのアクセスを平準化する事ができないという問題もあった。
【０００７】本発明は、上記従来の問題点に鑑み、分散配置されたｗｅｂサーバサイト間のアクセス負荷を平準化することを目的とする。
【０００８】また、分散配置されたｗｅｂサーバサイト間のアクセスを平準化するとともに、各々のｗｅｂサーバサイト内のｗｅｂサーバ間においてもアクセス負荷の平準化をすることを目的とする。
【０００９】また、最適なｗｅｂサーバを高速に決定するために、第一の決定手法でまず最適なｗｅｂサーバサイトを高速に決定し、次に第二の決定手法で真に最適なｗｅｂサーバを決定することで、できるだけ高速に最適なｗｅｂサーバを決定することを目的とする。
【００１０】また、最適なｗｅｂサーバを高速に決定するために、アクセスされたｗｅｂサーバがネットワーク状態、サーバ状態を高速に検知することを目的とする。
【００１１】また、アクセスクライアントが何ら能動的な操作の必要も無く最適ｗｅｂサーバに切り替えられることを目的とする。
【００１２】さらに、分散配置されたすべてのｗｅｂサーバによって最初のアクセスを受け付けることにより、アクセスの集中をなくす事を目的とする。
【００１３】
【課題を解決するための手段】請求項１に記載のネットワークステータスサーバは、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得手段と、前記取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定手段とを備えることを特徴とする。
【００１４】請求項２に記載のネットワークステータスサーバは請求項１に係るネットワークステータスサーバであって、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得手段は、前記論理的な距離を取得することを特徴とする。
【００１５】請求項３に記載のネットワークステータスサーバは請求項２に係るネットワークステータスサーバであって、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を蓄積する経路情報蓄積手段を備え、前記論理的距離取得手段は、前記経路情報蓄積手段に前記経路情報が蓄積されている場合、前記経路情報蓄積手段に蓄積されている経路情報から、前記論理的な距離を取得することを特徴とする。
【００１６】請求項４に記載のネットワークステータスサーバは請求項１乃至３のいずれかに係るネットワークステータスサーバであって、前記サイト決定手段は、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間の各々の論理的な距離が最小となるサイトを前記最適なサイトとして決定することを特徴とする。
【００１７】請求項５に記載のネットワークステータスサーバは請求項１乃至４のいずれかに係るネットワークステータスサーバであって、更に、前記サイト決定手段により決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定するサーバ決定手段を備えることを特徴とする。
【００１８】請求項６に記載のネットワークステータスサーバは請求項５に係るネットワークステータスサーバであって、前記サーバ決定手段は、前記最適なサイト内で予め決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする。
【００１９】請求項７に記載のネットワークステータスサーバは請求項５に係るネットワークステータスサーバであって、前記サーバ決定手段は、所定の順序で決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする。
【００２０】請求項８に記載のネットワークステータスサーバは請求項５に係るネットワークステータスサーバであって、前記サーバ決定手段は、前記最適なサイト内の情報配信サーバから任意に情報配信サーバを決定することを特徴とする。
【００２１】請求項９に記載のネットワークステータスサーバは請求項５に係るネットワークステータスサーバであって、更に、前記分散配置された情報配信サーバを有するサイト内の状態の情報を収集する状態情報収集手

段を有し、前記サーバ決定手段は、収集した前記最適なサイト内の状態の情報を条件として、最適な情報配信サーバを決定することを特徴とする。

【００２２】請求項１０に記載のネットワークステータスサーバは請求項９に係るネットワークステータスサーバであって、収集した前記サイト内の状態の情報を蓄積する状態情報蓄積手段とを有し、前記状態情報蓄積手段に前記最適なサイト内の状態の情報が蓄積されている場合、前記サーバ決定手段は、蓄積された前記最適なサイト内の状態の情報を条件として、最適な情報配信サーバを決定することを特徴とする。

【００２３】請求項１１に記載のネットワークステータスサーバは請求項１０に係るネットワークステータスサーバであって、前記状態情報蓄積手段は、前記蓄積したサイト内の状態の情報を所定の期間だけ蓄積することを特徴とする。

【００２４】請求項１２に記載のネットワークステータスサーバは請求項１０又は１１に係るネットワークステータスサーバであって、前記状態情報蓄積手段は、前記サイト内の状態の情報を所定の近似式で近似して蓄積することを特徴とする。

【００２５】請求項１３に記載のネットワークステータスサーバは請求項９乃至１２のいずれかに係るネットワークステータスサーバであって、前記サイト内の状態の情報とは、前記サイト内のネットワークの状態の情報と、前記サイト内の情報配信サーバの状態の情報との少なくともどちらか一つの情報であることを特徴とする。

【００２６】請求項１４に記載のネットワークステータスサーバは請求項１３に係るネットワークステータスサーバであって、前記分散配置された情報配信サーバを有する各サイト内のネットワークの状態の情報を収集する場合、前記収集手段は、前記各サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともどれか一つを収集することを特徴とする。

【００２７】請求項１５に記載のネットワークステータスサーバは請求項１３又は１４のいずれかに係るネットワークステータスサーバであって、前記分散配置された情報配信サーバの状態の情報を収集する収集手段は、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷の内、少なくともいずれか一つを収集することを特徴とする。

【００２８】請求項１６に記載のネットワークステータスサーバは、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得手段と、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集手段とを有し、前記論理的距離取得手段により取得した論理的な距離と、前記収集手段により収集した前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定するサーバ決定手段とを備えることを特徴とする。

【００２９】請求項１７に記載のネットワークステータスサーバは請求項１６に係るネットワークステータスサーバであって、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得手段は、前記論理的な距離を取得することを特徴とする。

【００３０】請求項１８に記載のネットワークステータスサーバは請求項１７に係るネットワークステータスサーバであって、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を蓄積する経路情報蓄積手段を備え、前記論理的距離取得手段は、前記経路情報蓄積手段に前記経路情報を蓄積している場合、前記経路情報蓄積手段に蓄積している経路情報から、前記論理的な距離を取得することを特徴とする。

【００３１】請求項１９に記載のネットワークステータスサーバは請求項１６乃至１８のいずれかに係るネットワークステータスサーバであって、更に、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを蓄積する状態情報蓄積手段を備え、前記収集手段は、前記状態情報蓄積手段に前記状態の情報を蓄積している場合、前記状態情報蓄積手段に蓄積している状態の情報から、前記状態の情報を収集することを特徴とする。

【００３２】請求項２０に記載のネットワークステータスサーバは請求項１９に係るネットワークステータスサーバであって、前記状態情報蓄積手段は、前記蓄積したサイト内の状態の情報を所定の期間だけ蓄積することを特徴とする。

【００３３】請求項２１に記載のネットワークステータスサーバは請求項１９又は２０のいずれかに係るネットワークステータスサーバであって、前記状態情報蓄積手段は、前記サイト内の状態の情報を所定の近似式で近似して蓄積する。

【００３４】請求項２２に記載のネットワークステータスサーバは請求項１６乃至２１のいずれかに係るネットワークステータスサーバであって、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報を収集する場合、前記収集手段は、前記アクセスしてきたクライアントと前記各サイトとの間の、応答時間、ルータ

段数、パケットロス率のうち、少なくともどれか一つを収集することを特徴とする。

【００３５】請求項２３に記載のネットワークステータスサーバは請求項１６乃至２２に係るネットワークステータスサーバであって、前記分散配置された情報配信サーバを有するサイト内の状態の情報とは、前記分散配置された情報配信サーバを有するサイト内のネットワークの状態の情報、又は、前記情報配信サーバの状態の情報の少なくともどちらか一つであることを特徴とする。

【００３６】請求項２４に記載のネットワークステータスサーバは請求項２３に係るネットワークステータスサーバであって、前記分散配置された情報配信サーバを有する各サイト内のネットワークの状態の情報を収集する場合、前記収集手段は、前記各サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともいずれか一つを収集することを特徴とする。

【００３７】請求項２５に記載のネットワークステータスサーバは請求項２３又は２４に係るネットワークステータスサーバであって、前記分散配置された情報配信サーバの状態の情報を収集する場合、前記収集手段は、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷の内、少なくともいずれか一つを収集することを特徴とする。

【００３８】請求項２６に記載のネットワークステータスサーバは請求項１６に係るネットワークステータスサーバであって、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を蓄積する経路情報蓄積手段と、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを蓄積する状態情報蓄積手段とを備え、前記サーバ決定手段は、蓄積された前記経路情報と前記状態の情報と所定の重み係数とに基づいて、下記の式；

・ネットワーク状態値：Ｋ１ｎ＝ＲＴｎ・Ａ＋ＲＮｎ・Ｂ＋ＰＬｎ・Ｃ

・サイト内ネットワーク状態値：Ｋ２ｎ＝ＣＳｎ・Ｄ＋ＰＳｎ・Ｅ＋ＥＳｎ・Ｆ

・サーバ状態値：Ｋ３ｎｍ＝ＣＰＵｎｍ・Ｇ＋ＩＤＬＥｎｍ・Ｈ＋ＬＩＮＫｎｍ

・Ｉ＋ＩＯｎｍ・Ｊ

・最適サイト判定値：Ｋｎ＝Ｋ１ｎ・Ｋ＋Ｋ２ｎ・Ｌ＋ＡＳＬｎ・Ｍ

・最適サーバ判定値：Ｓｎｍ＝Ｋｎ・Ｎ＋Ｋ３ｎｍ・Ｏ

（但し、ｎ：サーバサイト番号、ｍ：サーバ番号、Ａ～Ｏ：重み係数であり、各記号はそれぞれ、ＡＳＬｎ：ＡＳパスによる論理的なネットワーク間距離（ＢＧＰの経路情報を用いる）、ＲＴｎ：応答時間、ＲＮｎ：ルータ段数（ルータホップ数）、ＰＬｎ：パケットロス率、ＣＳｎ：サイト内の混雑度、ＰＳｎ：サイト内のパケット数、ＥＳｎ：パケットエラー、ＣＰＵｎｍ：ＣＰＵ負荷率、ＩＤＬＥｎｍ：ＣＰＵアイドル値、ＬＩＮＫｎｍ：コネクションリンク数、ＩＯｎｍ：ディスク負荷である。）における最適サーバ判定値Ｓｎｍが所定の条件を満たす情報配信サーバを最適な情報配信サーバとして決定することを特徴とする。

【００３９】請求項２７に記載のネットワークステータスサーバは、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得手段と、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集手段とを有し、前記クライアントからのアクセスが最初のアクセスであると判断した場合、前記論理的距離取得手段により取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定手段と、前記サイト決定手段により決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定する第１のサーバ決定手段とを備え、前記クライアントからのアクセスが最初のアクセスでないと判断した場合、前記論理的距離取得手段により取得した論理的な距離と、前記収集手段により収集した前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定する第２のサーバ決定手段とを備えることを特徴とする。

【００４０】請求項２８に記載のネットワークステータスサーバは請求項２７に係るネットワークステータスサーバであって、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得手段は、前記論理的な距離を取得することを特徴とする。

【００４１】請求項２９に記載のネットワークステータスサーバは請求項２８に係るネットワークステータスサーバであって、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を蓄積する経路情報蓄積手段を備え、前記論理的距離取得手段は、前記経路情報蓄積手段に前記経路情報を蓄積している場合、前記経路情報蓄積手段に蓄積している経路情報から、前記論理的な距離を取得することを特徴とする。

【００４２】請求項３０に記載のネットワークステータスサーバは請求項２７乃至２９のいずれかに係るネットワークステータスサーバであって、更に、前記分散配置

された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを蓄積する状態情報蓄積手段を備え、前記収集手段は、前記状態情報蓄積手段に前記状態の情報を蓄積している場合、前記状態情報蓄積手段に蓄積している状態の情報から、前記状態の情報を収集することを特徴とする。

【００４３】請求項３１に記載のネットワークステータスサーバは請求項３０に係るネットワークステータスサーバであって、前記状態情報蓄積手段は、前記蓄積したサイト内の状態の情報を所定の期間だけ蓄積することを特徴とする。

【００４４】請求項３２に記載のネットワークステータスサーバは請求項３０又は３１のいずれかに係るネットワークステータスサーバであって、前記状態情報蓄積手段は、前記サイト内の状態の情報を所定の近似式で近似して蓄積する。

【００４５】請求項３３に記載のネットワークステータスサーバは請求項２７乃至３２のいずれかに係るネットワークステータスサーバであって、前記サイト決定手段は、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間の各々の論理的な距離が最小となるサイトを前記最適なサイトとして決定することを特徴とする。

【００４６】請求項３４に記載のネットワークステータスサーバは請求項２７乃至３３のいずれかに係るネットワークステータスサーバであって、前記第１のサーバ決定手段は、前記最適なサイト内で予め決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする。

【００４７】請求項３５に記載のネットワークステータスサーバは請求項２７乃至３３のいずれかに係るネットワークステータスサーバであって、前記第１のサーバ決定手段は、所定の順序で決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする。

【００４８】請求項３６に記載のネットワークステータスサーバは請求項２７乃至３３のいずれかに係るネットワークステータスサーバであって、前記第１のサーバ決定手段は、前記最適なサイト内の情報配信サーバから任意に情報配信サーバを決定することを特徴とする。

【００４９】請求項３７に記載のネットワークステータスサーバは請求項２７乃至３３のいずれかに係るネットワークステータスサーバであって、前記第１のサーバ決定手段は、前記最適なサイト内の状態の情報に基づいて、最適な情報配信サーバを決定することを特徴とする。

【００５０】請求項３８に記載のネットワークステータスサーバは請求項２７乃至３７のいずれかに係るネットワークステータスサーバであって、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報を収集する場合、前記収集手段は、前記アクセスしてきたクライアントと前記各サイトとの間の、応答時間、ルータ段数、パケットロス率の内、少なくともどれか一つの情報を収集することを特徴とする。

【００５１】請求項３９に記載のネットワークステータスサーバは請求項２７乃至３８のいずれかに係るネットワークステータスサーバであって、前記サイト内の状態の情報とは、前記サイト内のネットワークの状態の情報と、前記サイト内の情報配信サーバの状態の情報との少なくともどちらか一つであることを特徴とする。

【００５２】請求項４０に記載のネットワークステータスサーバは請求項３９に係るネットワークステータスサーバであって、前記分散配置された情報配信サーバを有するサイト内のネットワークの状態の情報を収集する場合、前記収集手段は、前記サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともいずれか一つの情報を収集することを特徴とする。

【００５３】請求項４１に記載のネットワークステータスサーバは請求項３９又は４０のいずれかに係るネットワークステータスサーバであって、前記分散配置された情報配信サーバの状態の情報を収集する場合、前記収集手段は、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷のうち、少なくともどれか一つの情報を収集することを特徴とする。

【００５４】請求項４２に記載のネットワークステータスサーバは請求項２７に係るネットワークステータスサーバであって、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を蓄積する経路情報蓄積手段と、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを蓄積する状態情報蓄積手段とを備え、前記第２のサーバ決定手段は、蓄積された前記経路情報と前記状態の情報と所定の重み係数とに基づいて、下記の式；

・ネットワーク状態値：Ｋ１ｎ＝ＲＴｎ・Ａ＋ＲＮｎ・Ｂ＋ＰＬｎ・Ｃ

・サイト内ネットワーク状態値：Ｋ２ｎ＝ＣＳｎ・Ｄ＋ＰＳｎ・Ｅ＋ＥＳｎ・Ｆ

・サーバ状態値：Ｋ３ｎｍ＝ＣＰＵｎｍ・Ｇ＋ＩＤＬＥｎｍ・Ｈ＋ＬＩＮＫｎｍ

・Ｉ＋ＩＯｎｍ・Ｊ

・最適サイト判定値：Ｋｎ＝Ｋ１ｎ・Ｋ＋Ｋ２ｎ・Ｌ＋ＡＳＬｎ・Ｍ

・最適サーバ判定値：Ｓｎｍ＝Ｋｎ・Ｎ＋Ｋ３ｎｍ・Ｏ

（但し、ｎ：サーバサイト番号、ｍ：サーバ番号、Ａ～Ｏ：重み係数であり、各記号はそれぞれ、ＡＳＬｎ：ＡＳパスによる論理的なネットワーク間距離（ＢＧＰの経路情報を用いる）、ＲＴｎ：応答時間、ＲＮｎ：ルータ段数（ルータホップ数）、ＰＬｎ：パケットロス率、ＣＳｎ：サイト内の混雑度、ＰＳｎ：サイト内のパケット数、ＥＳｎ：パケットエラー、ＣＰＵｎｍ：ＣＰＵ負荷率、ＩＤＬＥｎｍ：ＣＰＵアイドル値、ＬＩＮＫｎｍ：コネクションリンク数、ＩＯｎｍ：ディスク負荷である。）における最適サーバ判定値Ｓｎｍが所定の条件を満たす情報配信サーバを最適な情報配信サーバとして決定することを特徴とする。

【００５５】請求項４３に記載の情報配信システムは、分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムであって、前記情報配信サーバは、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせる問い合せ手段と、前記ネットワークステータスサーバからの返答を受け取る返答受信手段と、前記アクセスするクライアントに前記最適な情報配信サーバを指示する指示手段とを備え、前記ネットワークステータスサーバは、前記問い合せ手段からの問い合せを受信する問い合せ受信手段と、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得手段と、前記取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定手段と、前記サイト決定手段により決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定するサーバ決定手段と、前記サーバ決定手段により、決定したサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答する返答送信手段とを備えることを特徴とする。

【００５６】請求項４４に記載の情報配信システムは請求項４３に係る情報配信システムであって、更に、前記情報配信システムはルートサーバを備え、前記ルートサーバはサイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有し、前記論理的距離取得手段が前記論理的な距離を取得する際に、前記論理的な距離を提供することを特徴とする。

【００５７】請求項４５に記載の情報配信システムは請求項４３又は４４のいずれかに係る情報配信システムであって、更に、前記情報配信サーバは、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集する手段と、収集した前記ネットワーク状態の情報をネットワークステータスサーバに送信するネットワーク状態送信手段とを備えることを特徴とする。

【００５８】請求項４６に記載の情報配信システムは請求項４３乃至４５のいずれかに係る情報配信システムであって、更に、前記情報配信サーバは、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集する手段と、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信するサイト内情報送信手段とを備えることを特徴とする。

【００５９】請求項４７に記載の情報配信システムは、分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムであって、前記情報配信サーバは、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせる問い合せ手段と、前記ネットワークステータスサーバからの返答を受け取る返答受信手段と、前記アクセスするクライアントに前記最適な情報配信サーバを指示する指示手段とを備え、前記ネットワークステータスサーバは、前記問い合せ手段からの問い合せを受信する問い合せ受信手段と、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得手段と、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集手段とを有し、前記論理的距離取得手段により取得した論理的な距離と、前記収集手段により収集した前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定するサーバ決定手段と、前記サーバ決定手段により、決定したサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答する返答送信手段とを備えることを特徴とする。

【００６０】請求項４８に記載の情報配信システムは請求項４７に係る情報配信システムであって、更に、前記情報配信システムはルートサーバを備え、前記ルートサーバはサイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有し、前記論理的距離取得手段が前記論理的な距離を取得する際に、前記論理的な距離を提供することを特徴とする。

【００６１】請求項４９に記載の情報配信システムは請求項４７又は４８のいずれかに係る情報配信システムであって、更に、前記情報配信サーバは、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集する手段と、収集した

前記ネットワーク状態の情報をネットワークステータスサーバに送信するネットワーク状態送信手段とを備えることを特徴とする。

【００６２】請求項５０に記載の情報配信システムは請求項４７乃至４９のいずれかに係る情報配信システムであって、更に、前記情報配信サーバは、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集する手段と、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信するサイト内情報送信手段とを備えることを特徴とする。

【００６３】請求項５１に記載の情報配信システムは、分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムであって、前記情報配信サーバは、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせる問い合せ手段と、前記ネットワークステータスサーバからの返答を受け取る返答受信手段と、前記アクセスするクライアントに前記最適な情報配信サーバを指示する指示手段とを備え、前記ネットワークステータスサーバは、前記問い合せ手段からの問い合せを受信する問い合せ受信手段と、前記分散配置された情報配信サーバを有する各サイトと前記情報配信サーバにアクセスするクライアントとの間の各々の論理的な距離を取得する論理的距離取得手段と、前記分散配置された情報配信サーバを有する各サイトと前記アクセスするクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集手段とを有し、前記クライアントからのアクセスが最初のアクセスであると判断した場合、前記論理的距離取得手段により取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定手段と、前記サイト決定手段により決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定する第１のサーバ決定手段とを有し、前記クライアントからのアクセスが最初のアクセスでないと判断した場合、前記論理的距離取得手段により取得した論理的な距離と、前記収集手段により収集した前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定する第２のサーバ決定手段とを有し、前記第１のサーバ決定手段もしくは前記第２のサーバ決定手段により、決定したサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答する返答送信手段とを備えることを特徴とする。

【００６４】請求項５２に記載の情報配信システムは請求項５１に係る情報配信システムであって、更に、前記情報配信システムはルートサーバを備え、前記ルートサーバはサイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有し、前記論理的距離取得手段が前記論理的な距離を取得する際に、前記論理的な距離を提供することを特徴とする。

【００６５】請求項５３に記載の情報配信システムは請求項５１又は５２のいずれかに係る情報配信システムであって、更に、前記情報配信サーバは、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集する手段と、収集した前記ネットワーク状態の情報をネットワークステータスサーバに送信するネットワーク状態送信手段とを備えることを特徴とする。

【００６６】請求項５４に記載の情報配信システムは請求項５１乃至５３のいずれかに係る情報配信システムであって、更に、前記情報配信サーバは、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集する手段と、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信するサイト内情報送信手段とを備えることを特徴とする。

【００６７】請求項５５に記載のネットワークステータスサーバ制御方法は、ネットワークステータスサーバを制御するネットワークステータスサーバ制御方法であって、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得ステップと、前記取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定ステップとを備えることを特徴とする。

【００６８】請求項５６に記載のネットワークステータスサーバ制御方法は請求項５５に係るネットワークステータスサーバ制御方法であって、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得ステップにおいて、前記論理的な距離を取得することを特徴とする。

【００６９】請求項５７に記載のネットワークステータスサーバ制御方法は請求項５６に係るネットワークステータスサーバ制御方法であって、前記ネットワークステータスサーバは経路情報蓄積手段を有し、前記ネットワークステータスサーバ制御方法は、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積する経路情報蓄積ステップを備え、前記論理的距離取得ステップでは、前記経路情報蓄積手段に前記

経路情報が蓄積されている場合、前記経路情報蓄積手段に蓄積されている経路情報から、前記論理的な距離を取得することを特徴とする。

【００７０】請求項５８に記載のネットワークステータスサーバ制御方法は請求項５５乃至５７のいずれかに係るネットワークステータスサーバ制御方法であって、前記サイト決定ステップでは、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間の各々の論理的な距離が最小となるサイトを前記最適なサイトとして決定することを特徴とする。

【００７１】請求項５９に記載のネットワークステータスサーバ制御方法は請求項５５乃至５８のいずれかに係るネットワークステータスサーバ制御方法であって、更に、前記サイト決定ステップにおいて決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定するサーバ決定ステップを備えることを特徴とする。

【００７２】請求項６０に記載のネットワークステータスサーバ制御方法は請求項５９に係るネットワークステータスサーバ制御方法であって、前記サーバ決定ステップでは、前記最適なサイト内で予め決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする。

【００７３】請求項６１に記載のネットワークステータスサーバ制御方法は請求項５９に係るネットワークステータスサーバ制御方法であって、前記サーバ決定ステップでは、所定の順序で決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする。

【００７４】請求項６２に記載のネットワークステータスサーバ制御方法は請求項５９に係るネットワークステータスサーバ制御方法であって、前記サーバ決定ステップでは、前記最適なサイト内の情報配信サーバから任意に情報配信サーバを決定することを特徴とする。

【００７５】請求項６３に記載のネットワークステータスサーバ制御方法は請求項５９に係るネットワークステータスサーバ制御方法であって、更に、前記分散配置された情報配信サーバを有するサイト内の状態の情報を収集する状態情報収集ステップを有し、前記サーバ決定ステップでは、収集した前記最適なサイト内の状態の情報を条件として、最適な情報配信サーバを決定することを特徴とする。

【００７６】請求項６４に記載のネットワークステータスサーバ制御方法は請求項６３に係るネットワークステータスサーバ制御方法であって、前記ネットワークステータスサーバは、状態情報蓄積手段を有し、前記ネットワークステータスサーバ制御方法は、更に、収集した前記サイト内の状態の情報を状態情報蓄積手段に蓄積する状態情報蓄積ステップを備え、前記状態情報蓄積手段に前記最適なサイト内の状態の情報が蓄積されている場合、前記サーバ決定ステップでは、蓄積された前記最適なサイト内の状態の情報を条件として、最適な情報配信サーバを決定することを特徴とする。

【００７７】請求項６５に記載のネットワークステータスサーバ制御方法は請求項６４に係るネットワークステータスサーバ制御方法であって、所定の期間を経過後、前記状態情報蓄積手段に蓄積したサイト内の状態の情報を破棄する状態情報破棄ステップを備えることを特徴とする。

【００７８】請求項６６に記載のネットワークステータスサーバ制御方法は請求項６４又は６５に係るネットワークステータスサーバ制御方法であって、前記状態情報蓄積ステップでは、前記サイト内の状態の情報を所定の近似式で近似して蓄積することを特徴とする。

【００７９】請求項６７に記載のネットワークステータスサーバ制御方法は請求項６３乃至６６のいずれかに係るネットワークステータスサーバ制御方法であって、前記サイト内の状態の情報とは、前記サイト内のネットワークの状態の情報と、前記サイト内の情報配信サーバの状態の情報との少なくともどちらか一つの情報であることを特徴とする。

【００８０】請求項６８に記載のネットワークステータスサーバ制御方法は請求項６７に係るネットワークステータスサーバ制御方法であって、前記分散配置された情報配信サーバを有する各サイト内のネットワークの状態の情報を収集する場合、前記収集ステップでは、前記各サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともどれか一つを収集することを特徴とする。

【００８１】請求項６９に記載のネットワークステータスサーバ制御方法は請求項６７又は６８のいずれかに係るネットワークステータスサーバ制御方法であって、前記分散配置された情報配信サーバの状態の情報を収集する収集ステップでは、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷の内、少なくともいずれか一つを収集することを特徴とする。

【００８２】請求項７０に記載のネットワークステータスサーバ制御方法は、ネットワークステータスサーバを制御するネットワークステータスサーバ制御方法であって、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集ステップと、前記論理的距離取得ステップにおいて取得した論理的な距離と、前記収集ステップにおいて収集した前記分散配置された情報配信サーバを有する各サイトと前記アクセスし

てきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定するサーバ決定ステップとを備えることを特徴とする。

【００８３】請求項７１に記載のネットワークステータスサーバ制御方法は請求項７０に係るネットワークステータスサーバ制御方法であって、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得ステップにおいて、前記論理的な距離を取得することを特徴とする。

【００８４】請求項７２に記載のネットワークステータスサーバ制御方法は請求項７１に係るネットワークステータスサーバ制御方法であって、前記ネットワークステータスサーバは経路情報蓄積手段を有し、前記ネットワークステータスサーバ制御方法は、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積する経路情報蓄積ステップを備え、前記論理的距離取得ステップでは、前記経路情報蓄積手段に前記経路情報を蓄積している場合、前記経路情報蓄積ステップに蓄積している経路情報から、前記論理的な距離を取得することを特徴とする。

【００８５】請求項７３に記載のネットワークステータスサーバ制御方法は請求項７０乃至７２のいずれかに係るネットワークステータスサーバ制御方法であって、前記ネットワークステータスサーバは状態情報蓄積手段を有し、前記ネットワークステータスサーバ制御方法は、更に、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを状態情報蓄積手段に蓄積する状態情報蓄積ステップを備え、前記収集ステップでは、前記状態情報蓄積手段に前記状態の情報を蓄積している場合、前記状態情報蓄積手段に蓄積している前記状態の情報から、前記状態の情報を収集することを特徴とする。

【００８６】請求項７４に記載のネットワークステータスサーバ制御方法は請求項７３に係るネットワークステータスサーバ制御方法であって、所定の期間を経過後、前記状態情報蓄積手段に蓄積したサイト内の状態の情報を破棄する状態情報破棄ステップを備えることを特徴とする。

【００８７】請求項７５に記載のネットワークステータスサーバ制御方法は請求項７３又は７４のいずれかに係るネットワークステータスサーバ制御方法であって、前記状態情報蓄積ステップでは、前記サイト内の状態の情報を所定の近似式で近似して蓄積する。

【００８８】請求項７６に記載のネットワークステータスサーバ制御方法は請求項７０乃至７５のいずれかに係るネットワークステータスサーバ制御方法であって、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報を収集する場合、前記収集ステップでは、前記アクセスしてきたクライアントと前記各サイトとの間の、応答時間、ルータ段数、パケットロス率のうち、少なくともどれか一つを収集することを特徴とする。

【００８９】請求項７７に記載のネットワークステータスサーバ制御方法は請求項７０乃至７６のいずれかに係るネットワークステータスサーバ制御方法であって、前記分散配置された情報配信サーバを有するサイト内の状態の情報とは、前記分散配置された情報配信サーバを有するサイト内のネットワークの状態の情報、又は、前記情報配信サーバの状態の情報の少なくともどちらか一つであることを特徴とする。

【００９０】請求項７８に記載のネットワークステータスサーバ制御方法は請求項７７に係るネットワークステータスサーバ制御方法であって、前記分散配置された情報配信サーバを有する各サイト内のネットワークの状態の情報を収集する場合、前記収集ステップでは、前記各サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともいずれか一つを収集することを特徴とする。

【００９１】請求項７９に記載のネットワークステータスサーバ制御方法は請求項７７又は７８に係るネットワークステータスサーバ制御方法であって、前記分散配置された情報配信サーバの状態の情報を収集する場合、前記収集ステップでは、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷の内、少なくともいずれか一つを収集することを特徴とする。

【００９２】請求項８０に記載のネットワークステータスサーバ制御方法は請求項７０に係るネットワークステータスサーバ制御方法であって、前記ネットワークステータスサーバは、経路情報蓄積手段と状態情報蓄積手段とを有し、前記ネットワークステータスサーバ制御方法は、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を経路情報蓄積手段に蓄積する経路情報蓄積ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを状態情報蓄積手段に蓄積する状態情報蓄積ステップとを備え、前記サーバ決定ステップでは、蓄積された前記経路情報と前記状態の情報と所定の重み係数とに基づいて、下記の式；

・ネットワーク状態値：Ｋ１ｎ＝ＲＴｎ・Ａ＋ＲＮｎ・Ｂ＋ＰＬｎ・Ｃ

・サイト内ネットワーク状態値：Ｋ２ｎ＝ＣＳｎ・Ｄ＋

ＰＳｎ・Ｅ＋ＥＳｎ・Ｆ
・サーバ状態値：Ｋ３ｎｍ＝ＣＰＵｎｍ・Ｇ＋ＩＤＬＥｎｍ・Ｈ＋ＬＩＮＫｎｍ
・Ｉ＋ＩＯｎｍ・Ｊ
・最適サイト判定値：Ｋｎ＝Ｋ１ｎ・Ｋ＋Ｋ２ｎ・Ｌ＋ＡＳＬｎ・Ｍ
・最適サーバ判定値：Ｓｎｍ＝Ｋｎ・Ｎ＋Ｋ３ｎｍ・Ｏ
（但し、ｎ：サーバサイト番号、ｍ：サーバ番号、Ａ～Ｏ：重み係数であり、各記号はそれぞれ、ＡＳＬｎ：ＡＳパスによる論理的なネットワーク間距離（ＢＧＰの経路情報を用いる）、ＲＴｎ：応答時間、ＲＮｎ：ルータ段数（ルータホップ数）、ＰＬｎ：パケットロス率、ＣＳｎ：サイト内の混雑度、ＰＳｎ：サイト内のパケット数、ＥＳｎ：パケットエラー、ＣＰＵｎｍ：ＣＰＵ負荷率、ＩＤＬＥｎｍ：ＣＰＵアイドル値、ＬＩＮＫｎｍ：コネクションリンク数、ＩＯｎｍ：ディスク負荷である。）における最適サーバ判定値Ｓｎｍが所定の条件を満たす情報配信サーバを最適な情報配信サーバとして決定することを特徴とする。

【００９３】請求項８１に記載のネットワークステータスサーバ制御方法は、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集ステップとを有し、前記クライアントからのアクセスが最初のアクセスであると判断した場合、前記論理的距離取得ステップにおいて取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定ステップと、前記サイト決定ステップにおいて決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定する第１のサーバ決定ステップとを備え、前記クライアントからのアクセスが最初のアクセスでないと判断した場合、前記論理的距離取得ステップにおいて取得した論理的な距離と、前記収集ステップにおいて収集した前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定する第２のサーバ決定ステップとを備えることを特徴とする。

【００９４】請求項８２に記載のネットワークステータスサーバ制御方法は請求項８１に係るネットワークステータスサーバ制御方法であって、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得ステップにおいて、前記論理的な距離を取得することを特徴とする。

【００９５】請求項８３に記載のネットワークステータスサーバ制御方法は請求項８２に係るネットワークステータスサーバ制御方法であって、前記ネットワークステータスサーバは経路情報蓄積手段を有し、前記ネットワークステータスサーバ制御方法は、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積する経路情報蓄積ステップを備え、前記論理的距離取得ステップでは、前記経路情報蓄積手段に前記経路情報を蓄積している場合、前記経路情報蓄積手段に蓄積している経路情報から、前記論理的な距離を取得することを特徴とする。

【００９６】請求項８４に記載のネットワークステータスサーバ制御方法は請求項８１乃至８３のいずれかに係るネットワークステータスサーバ制御方法であって、前記ネットワークステータスサーバは状態情報蓄積手段を有し、前記ネットワークステータスサーバ制御方法は、更に、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを前記状態情報蓄積手段に蓄積する状態情報蓄積ステップを備え、前記収集ステップでは、前記状態情報蓄積手段に前記状態の情報を蓄積している場合、前記状態情報蓄積手段に蓄積している前記状態の情報から、前記状態の情報を収集することを特徴とする。

【００９７】請求項８５に記載のネットワークステータスサーバ制御方法は請求項８４に係るネットワークステータスサーバ制御方法であって、所定の期間を経過後、前記状態情報蓄積手段に蓄積したサイト内の状態の情報を破棄する状態情報破棄ステップを備えることを特徴とする。

【００９８】請求項８６に記載のネットワークステータスサーバ制御方法は請求項８４又は８５のいずれかに係るネットワークステータスサーバ制御方法であって、前記状態情報蓄積ステップでは、前記サイト内の状態の情報を所定の近似式で近似して蓄積する。

【００９９】請求項８７に記載のネットワークステータスサーバ制御方法は請求項８１乃至８６のいずれかに係るネットワークステータスサーバ制御方法であって、前記サイト決定ステップでは、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間の各々の論理的な距離が最小となるサイトを前記最適なサイトとして決定することを特徴とする。

【０１００】請求項８８に記載のネットワークステータスサーバ制御方法は請求項８１乃至８７のいずれかに係るネットワークステータスサーバ制御方法であって、前

記第１のサーバ決定ステップでは、前記最適なサイト内で予め決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする。

【０１０１】請求項８９に記載のネットワークステータスサーバ制御方法は請求項８１乃至８７のいずれかに係るネットワークステータスサーバ制御方法であって、前記第１のサーバ決定ステップでは、所定の順序で決められた情報配信サーバを、最適な情報配信サーバとして決定することを特徴とする。

【０１０２】請求項９０に記載のネットワークステータスサーバ制御方法は請求項８１乃至８７のいずれかに係るネットワークステータスサーバ制御方法であって、前記第１のサーバ決定ステップでは、前記最適なサイト内の情報配信サーバから任意に情報配信サーバを決定することを特徴とする。

【０１０３】請求項９１に記載のネットワークステータスサーバ制御方法は請求項８１乃至８７のいずれかに係るネットワークステータスサーバ制御方法であって、前記第１のサーバ決定ステップでは、前記最適なサイト内の状態の情報に基づいて、最適な情報配信サーバを決定することを特徴とする。

【０１０４】請求項９２に記載のネットワークステータスサーバ制御方法は請求項８１乃至９１のいずれかに係るネットワークステータスサーバ制御方法であって、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報を収集する場合、前記収集ステップでは、前記アクセスしてきたクライアントと前記各サイトとの間の、応答時間、ルータ段数、パケットロス率の内、少なくともどれか一つの情報を収集することを特徴とする。

【０１０５】請求項９３に記載のネットワークステータスサーバ制御方法は請求項８１乃至９２のいずれかに係るネットワークステータスサーバ制御方法であって、前記サイト内の状態の情報とは、前記サイト内のネットワークの状態の情報と、前記サイト内の情報配信サーバの状態の情報との少なくともどちらか一つであることを特徴とする。

【０１０６】請求項９４に記載のネットワークステータスサーバ制御方法は請求項９３に係るネットワークステータスサーバ制御方法であって、前記分散配置された情報配信サーバを有するサイト内のネットワークの状態の情報を収集する場合、前記収集ステップでは、前記サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともいずれか一つの情報を収集することを特徴とする。

【０１０７】請求項９５に記載のネットワークステータスサーバ制御方法は請求項９３又は９４のいずれかに係るネットワークステータスサーバ制御方法であって、前記分散配置された情報配信サーバの状態の情報を収集する場合、前記収集ステップでは、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷のうち、少なくともどれか一つの情報を収集することを特徴とする。

【０１０８】請求項９６に記載のネットワークステータスサーバ制御方法は請求項８１に係るネットワークステータスサーバ制御方法であって、前記ネットワークステータスサーバは、経路情報蓄積手段と状態情報蓄積手段とを有し、前記ネットワークステータスサーバ制御方法は、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積する経路情報蓄積ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを前記状態情報蓄積手段に蓄積する状態情報蓄積ステップとを備え、前記第２のサーバ決定ステップでは、蓄積された前記経路情報と前記状態の情報と所定の重み係数とに基づいて、下記の式；

・ネットワーク状態値：Ｋ１ｎ＝ＲＴｎ・Ａ＋ＲＮｎ・Ｂ＋ＰＬｎ・Ｃ

・サイト内ネットワーク状態値：Ｋ２ｎ＝ＣＳｎ・Ｄ＋ＰＳｎ・Ｅ＋ＥＳｎ・Ｆ

・サーバ状態値：Ｋ３ｎｍ＝ＣＰＵｎｍ・Ｇ＋ＩＤＬＥｎｍ・Ｈ＋ＬＩＮＫｎｍ

・Ｉ＋ＩＯｎｍ・Ｊ

・最適サイト判定値：Ｋｎ＝Ｋ１ｎ・Ｋ＋Ｋ２ｎ・Ｌ＋ＡＳＬｎ・Ｍ

・最適サーバ判定値：Ｓｎｍ＝Ｋｎ・Ｎ＋Ｋ３ｎｍ・Ｏ

（但し、ｎ：サーバサイト番号、ｍ：サーバ番号、Ａ～Ｏ：重み係数であり、各記号はそれぞれ、ＡＳＬｎ：ＡＳパスによる論理的なネットワーク間距離（ＢＧＰの経路情報を用いる）、ＲＴｎ：応答時間、ＲＮｎ：ルータ段数（ルータホップ数）、ＰＬｎ：パケットロス率、ＣＳｎ：サイト内の混雑度、ＰＳｎ：サイト内のパケット数、ＥＳｎ：パケットエラー、ＣＰＵｎｍ：ＣＰＵ負荷率、ＩＤＬＥｎｍ：ＣＰＵアイドル値、ＬＩＮＫｎｍ：コネクションリンク数、ＩＯｎｍ：ディスク負荷である。）における最適サーバ判定値Ｓｎｍが所定の条件を満たす情報配信サーバを最適な情報配信サーバとして決定することを特徴とする。

【０１０９】請求項９７に記載の情報配信システム制御方法は、分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムを制御する情報配信システム制御方法であって、前記情報配信サーバにおいて、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせる問い合せステップと、前記ネットワークステー

タスサーバからの返答を受け取る返答受信ステップと、前記アクセスするクライアントに前記最適な情報配信サーバを指示する指示ステップとを備え、前記ネットワークステータスサーバにおいて、前記問い合せステップからの問い合せを受信する問い合せ受信ステップと、分散配置された情報配信サーバを有する各サイトとアクセスするクライアントとの間の各々の論理的な距離を取得する論理的距離取得ステップと、前記取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定ステップと、前記サイト決定ステップにより決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定するサーバ決定ステップと、前記サーバ決定ステップにおいて決定したサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答する返答送信ステップとを備えることを特徴とする。

【０１１０】請求項９８に記載の情報配信システム制御方法は請求項９７に係る情報配信システム制御方法であって、更に、前記情報配信システムはルートサーバを有し、前記ルートサーバにおいて、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める論理的距離算出ステップを備え、前記論理的距離取得ステップでは、前記論理的距離算出ステップで算出した前記論理的な距離を取得することを特徴とする。

【０１１１】請求項９９に記載の情報配信システム制御方法は請求項９７又は９８のいずれかに係る情報配信システム制御方法であって、更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集するステップと、収集した前記ネットワーク状態の情報をネットワークステータスサーバに送信するネットワーク状態送信ステップとを備えることを特徴とする。

【０１１２】請求項１００に記載の情報配信システム制御方法は請求項９７乃至９９のいずれかに係る情報配信システム制御方法であって、更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集するステップと、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信するサイト内情報送信ステップとを備えることを特徴とする。

【０１１３】請求項１０１に記載の情報配信システム制御方法は、分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムを制御する情報配信システム制御方法であって、前記情報配信サーバにおいて、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせる問い合せステップと、前記ネットワークステータスサーバからの返答を受け取る返答受信ステップと、前記アクセスするクライアントに前記最適な情報配信サーバを指示する指示ステップとを備え、前記ネットワークステータスサーバにおいて、前記問い合せステップからの問い合せを受信する問い合せ受信ステップと、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集ステップと、前記論理的距離取得ステップにおいて取得した論理的な距離と、前記収集ステップにおいて収集した、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定するサーバ決定ステップと、前記サーバ決定ステップにおいて決定したサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答する返答送信ステップとを備えることを特徴とする。

【０１１４】請求項１０２に記載の情報配信システム制御方法は請求項１０１に係る情報配信システム制御方法であって、更に、前記情報配信システムはルートサーバを有し、前記ルートサーバにおいて、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める論理的距離算出ステップを備え、前記論理的距離取得ステップでは、前記論理的距離算出ステップで算出した前記論理的な距離を取得することを特徴とする。

【０１１５】請求項１０３に記載の情報配信システム制御方法は請求項１０１又は１０２に係る情報配信システム制御方法であって、更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集するステップと、収集した前記ネットワーク状態の情報をネットワークステータスサーバに送信するネットワーク状態送信ステップとを備えることを特徴とする。

【０１１６】請求項１０４に記載の情報配信システム制御方法は請求項１０１乃至１０３のいずれかに係る情報配信システム制御方法であって、更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集するステップと、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信するサイト内情報送信ステップとを備えることを特徴とする。

【０１１７】請求項１０５に記載の情報配信システム制

御方法は、分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムを制御する情報配信システム制御方法であって、前記情報配信サーバにおいて、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせる問い合せステップと、前記ネットワークステータスサーバからの返答を受け取る返答受信ステップと、前記アクセスするクライアントに前記最適な情報配信サーバを指示する指示ステップとを備え、前記ネットワークステータスサーバにおいて、前記問い合せステップからの問い合せを受信する問い合せ受信ステップと、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得する論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集する収集ステップと、前記クライアントからのアクセスが最初のアクセスであると判断した場合、前記論理的距離取得ステップにおいて取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定するサイト決定ステップと、前記サイト決定ステップにおいて決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定する第１のサーバ決定ステップと、前記クライアントからのアクセスが最初のアクセスでないと判断した場合、前記論理的距離取得ステップにおいて取得した論理的な距離と、前記収集ステップにおいて収集した、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定する第２のサーバ決定ステップと、前記第１のサーバ決定ステップもしくは前記第２のサーバ決定ステップにおいて決定したサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答する返答送信ステップとを備えることを特徴とする。

【０１１８】請求項１０６に記載の情報配信システム制御方法は請求項１０５に係る情報配信システム制御方法であって、更に、前記情報配信システムはルートサーバを備え、前記ルートサーバにおいて、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める論理的距離算出ステップを備え、前記論理的距離取得ステップでは、前記論理的距離算出ステップで算出した前記論理的な距離を取得することを特徴とする。

【０１１９】請求項１０７に記載の情報配信システム制御方法は請求項１０５又は１０６のいずれかに係る情報配信システム制御方法であって、更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集するステップと、収集した前記ネットワーク状態の情報をネットワークステータスサーバに送信するネットワーク状態送信ステップとを備えることを特徴とする。

【０１２０】請求項１０８に記載の情報配信システム制御方法は請求項１０５乃至１０７のいずれかに係る情報配信システム制御方法であって、更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集するステップと、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信するサイト内情報送信ステップとを備えることを特徴とする。

【０１２１】請求項１０９に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体はネットワークステータスサーバを制御するコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体であって、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得させる論理的距離取得ステップと、前記取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定させるサイト決定ステップとを備えることを特徴とする。

【０１２２】請求項１１０に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１０９に係る記憶媒体であって、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得ステップにおいて、前記論理的な距離を取得させることを特徴とする。

【０１２３】請求項１１１に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１１０に係る記憶媒体であって、前記ネットワークステータスサーバは経路情報蓄積手段を有し、前記ネットワークステータスサーバ制御プログラムは、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積させる経路情報蓄積ステップを備え、前記論理的距離取得ステップでは、前記経路情報蓄積手段に前記経路情報が蓄積されている場合、前記経路情報蓄積手段に蓄積されている経路情報から、前記論理的な距離を取得させることを特徴とする。

【０１２４】請求項１１２に記載のコンピュータ読み取

り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１０９乃至１１１のいずれかに係る記憶媒体であって、前記サイト決定ステップでは、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間の各々の論理的な距離が最小となるサイトを前記最適なサイトとして決定させることを特徴とする。

【０１２５】請求項１１３に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１０９乃至１１２のいずれかに係る記憶媒体であって、更に、前記サイト決定ステップにおいて決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定させるサーバ決定ステップを備えることを特徴とする。

【０１２６】請求項１１４に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１１３に係る記憶媒体であって、前記サーバ決定ステップでは、前記最適なサイト内で予め決められた情報配信サーバを、最適な情報配信サーバとして決定させることを特徴とする。

【０１２７】請求項１１５に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１１３に係る記憶媒体であって、前記サーバ決定ステップでは、所定の順序で決められた情報配信サーバを、最適な情報配信サーバとして決定させることを特徴とする。

【０１２８】請求項１１６に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１１３に係る記憶媒体であって、前記サーバ決定ステップでは、前記最適なサイト内の情報配信サーバから任意に情報配信サーバを決定させることを特徴とする。

【０１２９】請求項１１７に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１１３に係る記憶媒体であって、更に、前記分散配置された情報配信サーバを有するサイト内の状態の情報を収集させる状態情報収集ステップを有し、前記サーバ決定ステップでは、収集した前記最適なサイト内の状態の情報を条件として、最適な情報配信サーバを決定させることを特徴とする。

【０１３０】請求項１１８に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１１７に係る記憶媒体であって、前記ネットワークステータスサーバは、状態情報蓄積手段を有し、前記ネットワークステータスサーバ制御プログラムは、更に、収集した前記サイト内の状態の情報を状態情報蓄積手段に蓄積させる状態情報蓄積ステップを備え、前記状態情報蓄積手段に前記最適なサイト内の状態の情報が蓄積されている場合、前記サーバ決定ステップでは、蓄積された前記最適なサイト内の状態の情報を条件として、最適な情報配信サーバを決定させることを特徴とする。

【０１３１】請求項１１９に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１１８に係る記憶媒体であって、所定の期間を経過後、前記状態情報蓄積手段に蓄積したサイト内の状態の情報を破棄させる状態情報破棄ステップを備えることを特徴とする。

【０１３２】請求項１２０に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１１８又は１１９に係る記憶媒体であって、前記状態情報蓄積ステップでは、前記サイト内の状態の情報を所定の近似式で近似して蓄積させることを特徴とする。

【０１３３】請求項１２１に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１１７乃至１２０のいずれかに係る記憶媒体であって、前記サイト内の状態の情報とは、前記サイト内のネットワークの状態の情報と、前記サイト内の情報配信サーバの状態の情報との少なくともどちらか一つの情報であることを特徴とする。

【０１３４】請求項１２２に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１２１に係る記憶媒体であって、前記分散配置された情報配信サーバを有する各サイト内のネットワークの状態の情報を収集させる場合、前記収集ステップでは、前記各サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともどれか一つを収集させることを特徴とする。

【０１３５】請求項に１２３記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１２１又は１２２のいずれかに係る記憶媒体であって、前記分散配置された情報配信サーバの状態の情報を収集させる収集ステップでは、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷の内、少なくともいずれか一つを収集させることを特徴とする。

【０１３６】請求項１２４に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は、ネットワークステータスサーバを制御するコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体であって、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得させる論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集させる収集ステップ

と、前記論理的距離取得ステップにおいて取得させた論理的な距離と、前記収集ステップにおいて収集させた、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定させるサーバ決定ステップとを備えることを特徴とする。

【０１３７】請求項１２５に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１２４に係る記憶媒体であって、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得ステップにおいて、前記論理的な距離を取得させることを特徴とする。

【０１３８】請求項１２６に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１２５に係る記憶媒体であって、前記ネットワークステータスサーバは経路情報蓄積手段を有し、前記ネットワークステータスサーバ制御プログラムは、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積させる経路情報蓄積ステップを備え、前記論理的距離取得ステップでは、前記経路情報蓄積手段に前記経路情報を蓄積している場合、前記経路情報蓄積ステップに蓄積している経路情報から、前記論理的な距離を取得させることを特徴とする。

【０１３９】請求項１２７に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１２４乃至１２６のいずれかに係る記憶媒体であって、前記ネットワークステータスサーバは状態情報蓄積手段を有し、前記ネットワークステータスサーバ制御プログラムは、更に、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを状態情報蓄積手段に蓄積させる状態情報蓄積ステップを備え、前記収集ステップでは、前記状態情報蓄積手段に前記状態の情報を蓄積している場合、前記状態情報蓄積手段に蓄積している前記状態の情報から、前記状態の情報を収集させることを特徴とする。

【０１４０】請求項１２８に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１２７に係る記憶媒体であって、所定の期間を経過後、前記状態情報蓄積手段に蓄積したサイト内の状態の情報を破棄させる状態情報破棄ステップを備えることを特徴とする。

【０１４１】請求項１２９に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１２７又は１２８のいずれかに係る記憶媒体であって、前記状態情報蓄積ステップでは、前記サイト内の状態の情報を所定の近似式で近似して蓄積させる。

【０１４２】請求項１３０に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１２４乃至１２９のいずれかに係る記憶媒体であって、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報を収集させる場合、前記収集ステップでは、前記アクセスしてきたクライアントと前記各サイトとの間の、応答時間、ルータ段数、パケットロス率のうち、少なくともどれか一つを収集させることを特徴とする。

【０１４３】請求項１３１に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１２４乃至１３０のいずれかに係る記憶媒体であって、前記分散配置された情報配信サーバを有するサイト内の状態の情報とは、前記分散配置された情報配信サーバを有するサイト内のネットワークの状態の情報、又は、前記情報配信サーバの状態の情報の少なくともどちらか一つであることを特徴とする。

【０１４４】請求項１３２に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３１に係る記憶媒体であって、前記分散配置された情報配信サーバを有する各サイト内のネットワークの状態の情報を収集させる場合、前記収集ステップでは、前記各サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともいずれか一つを収集させることを特徴とする。

【０１４５】請求項１３３に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３１又は１３２に係る記憶媒体であって、前記分散配置された情報配信サーバの状態の情報を収集させる場合、前記収集ステップでは、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷の内、少なくともいずれか一つを収集させることを特徴とする。

【０１４６】請求項１３４に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１２４に係る記憶媒体であって、前記ネットワークステータスサーバは、経路情報蓄積手段と状態情報蓄積手段とを有し、前記ネットワークステータスサーバ制御プログラムは、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を経路情報蓄積手段に蓄積させる経路情報蓄積ステップと、前記分散配置され

た情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを状態情報蓄積手段に蓄積させる状態情報蓄積ステップとを備え、前記サーバ決定ステップでは、蓄積された前記経路情報と前記状態の情報と所定の重み係数とに基づいて、下記の式；

・ネットワーク状態値：Ｋ１ｎ＝ＲＴｎ・Ａ＋ＲＮｎ・Ｂ＋ＰＬｎ・Ｃ

・サイト内ネットワーク状態値：Ｋ２ｎ＝ＣＳｎ・Ｄ＋ＰＳｎ・Ｅ＋ＥＳｎ・Ｆ

・サーバ状態値：Ｋ３ｎｍ＝ＣＰＵｎｍ・Ｇ＋ＩＤＬＥｎｍ・Ｈ＋ＬＩＮＫｎｍ

・Ｉ＋ＩＯｎｍ・Ｊ

・最適サイト判定値：Ｋｎ＝Ｋ１ｎ・Ｋ＋Ｋ２ｎ・Ｌ＋ＡＳＬｎ・Ｍ

・最適サーバ判定値：Ｓｎｍ＝Ｋｎ・Ｎ＋Ｋ３ｎｍ・Ｏ

（但し、ｎ：サーバサイト番号、ｍ：サーバ番号、Ａ～Ｏ：重み係数であり、各記号はそれぞれ、ＡＳＬｎ：ＡＳパスによる論理的なネットワーク間距離（ＢＧＰの経路情報を用いる）、ＲＴｎ：応答時間、ＲＮｎ：ルータ段数（ルータホップ数）、ＰＬｎ：パケットロス率、ＣＳｎ：サイト内の混雑度、ＰＳｎ：サイト内のパケット数、ＥＳｎ：パケットエラー、ＣＰＵｎｍ：ＣＰＵ負荷率、ＩＤＬＥｎｍ：ＣＰＵアイドル値、ＬＩＮＫｎｍ：コネクションリンク数、ＩＯｎｍ：ディスク負荷である。）における最適サーバ判定値Ｓｎｍが所定の条件を満たす情報配信サーバを最適な情報配信サーバとして決定させることを特徴とする。

【０１４７】請求項１３５に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は、分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得させる論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集させる収集ステップとを有し、前記クライアントからのアクセスが最初のアクセスであると判断した場合、前記論理的距離取得ステップにおいて取得させた論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定させるサイト決定ステップと、前記サイト決定ステップにおいて決定させた最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定させる第１のサーバ決定ステップとを備え、前記クライアントからのアクセスが最初のアクセスでないと判断した場合、前記論理的距離取得ステップにおいて取得させた論理的な距離と、前記収集ステップにおいて収集させた前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定させる第２のサーバ決定ステップとを備えることを特徴とする。

【０１４８】請求項１３６に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３５に係る記憶媒体であって、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める手段を有するルートサーバから、前記論理的距離取得ステップにおいて、前記論理的な距離を取得させることを特徴とする。

【０１４９】請求項１３７に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３６に係る記憶媒体であって、前記ネットワークステータスサーバは経路情報蓄積手段を有し、前記ネットワークステータスサーバ制御プログラムは、更に、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積させる経路情報蓄積ステップを備え、前記論理的距離取得ステップでは、前記経路情報蓄積手段に前記経路情報を蓄積している場合、前記経路情報蓄積手段に蓄積している経路情報から、前記論理的な距離を取得させることを特徴とする。

【０１５０】請求項１３８に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３５乃至１３７のいずれかに係る記憶媒体であって、前記ネットワークステータスサーバは状態情報蓄積手段を有し、前記ネットワークステータスサーバ制御プログラムは、更に、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを前記状態情報蓄積手段に蓄積させる状態情報蓄積ステップを備え、前記収集ステップでは、前記状態情報蓄積手段に前記状態の情報を蓄積している場合、前記状態情報蓄積手段に蓄積している前記状態の情報から、前記状態の情報を収集させることを特徴とする。

【０１５１】請求項１３９に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３８に係る記憶媒体であって、所定の期間を経過後、前記状態情報蓄積手段に蓄積したサイト内の状態の情報を破棄させる状態情報破棄ステップを備えることを特徴とする。

【０１５２】請求項１４０に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３８又は１３９のいずれ

かに係る記憶媒体であって、前記状態情報蓄積ステップでは、前記サイト内の状態の情報を所定の近似式で近似して蓄積させる。

【０１５３】請求項１４１に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３５乃至１４０のいずれかに係る記憶媒体であって、前記サイト決定ステップでは、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間の各々の論理的な距離が最小となるサイトを前記最適なサイトとして決定させることを特徴とする。

【０１５４】請求項１４２に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３５乃至１４１のいずれかに係る記憶媒体であって、前記第１のサーバ決定ステップでは、前記最適なサイト内で予め決められた情報配信サーバを、最適な情報配信サーバとして決定させることを特徴とする。

【０１５５】請求項１４３に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３５乃至１４１のいずれかに係る記憶媒体であって、前記第１のサーバ決定ステップでは、所定の順序で決められた情報配信サーバを、最適な情報配信サーバとして決定させることを特徴とする。

【０１５６】請求項１４４に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３５乃至１４１のいずれかに係る記憶媒体であって、前記第１のサーバ決定ステップでは、前記最適なサイト内の情報配信サーバから任意に情報配信サーバを決定させることを特徴とする。

【０１５７】請求項１４５に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３５乃至１４１のいずれかに係る記憶媒体であって、前記第１のサーバ決定ステップでは、前記最適なサイト内の状態の情報に基づいて、最適な情報配信サーバを決定させることを特徴とする。

【０１５８】請求項１４６に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３５乃至１４５のいずれかに係る記憶媒体であって、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報を収集させる場合、前記収集ステップでは、前記アクセスしてきたクライアントと前記各サイトとの間の、応答時間、ルータ段数、パケットロス率の内、少なくともどれか一つの情報を収集させることを特徴とする。

【０１５９】請求１４７に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３５乃至１４６のいずれかに係る記憶媒体であって、前記サイト内の状態の情報とは、前記サイト内のネットワークの状態の情報と、前記サイト内の情報配信サーバの状態の情報との少なくともどちらか一つであることを特徴とする。

【０１６０】請求項１４８に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１４７に係る記憶媒体であって、前記分散配置された情報配信サーバを有するサイト内のネットワークの状態の情報を収集させる場合、前記収集ステップでは、前記サイト内の、混雑度、パケット数、パケットエラー数のうち、少なくともいずれか一つの情報を収集させることを特徴とする。

【０１６１】請求項１４９に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１４７又は１４８のいずれかに係る記憶媒体であって、前記分散配置された情報配信サーバの状態の情報を収集させる場合、前記収集ステップでは、前記情報配信サーバの、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷のうち、少なくともどれか一つの情報を収集させることを特徴とする。

【０１６２】請求項１５０に記載のコンピュータ読み取り可能なネットワークステータスサーバ制御プログラムを格納した記憶媒体は請求項１３５に係る記憶媒体であって、前記ネットワークステータスサーバは、経路情報蓄積手段と状態情報蓄積手段とを有し、前記ネットワークステータスサーバ制御プログラムは、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の経路情報を前記経路情報蓄積手段に蓄積させる経路情報蓄積ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを前記状態情報蓄積手段に蓄積させる状態情報蓄積ステップとを備え、前記第２のサーバ決定ステップでは、蓄積された前記経路情報と前記状態の情報と所定の重み係数とに基づいて、下記の式；

・ネットワーク状態値：Ｋ１ｎ＝ＲＴｎ・Ａ＋ＲＮｎ・Ｂ＋ＰＬｎ・Ｃ

・サイト内ネットワーク状態値：Ｋ２ｎ＝ＣＳｎ・Ｄ＋ＰＳｎ・Ｅ＋ＥＳｎ・Ｆ

・サーバ状態値：Ｋ３ｎｍ＝ＣＰＵｎｍ・Ｇ＋ＩＤＬＥｎｍ・Ｈ＋ＬＩＮＫｎｍ

・Ｉ＋ＩＯｎｍ・Ｊ

・最適サイト判定値：Ｋｎ＝Ｋ１ｎ・Ｋ＋Ｋ２ｎ・Ｌ＋ＡＳＬｎ・Ｍ

・最適サーバ判定値：Ｓｎｍ＝Ｋｎ・Ｎ＋Ｋ３ｎｍ・Ｏ

（但し、ｎ：サーバサイト番号、ｍ：サーバ番号、Ａ～Ｏ：重み係数であり、各記号はそれぞれ、ＡＳＬｎ：Ａ

Ｓパスによる論理的なネットワーク間距離（ＢＧＰの経路情報を用いる）、ＲＴｎ：応答時間、ＲＮｎ：ルータ段数（ルータホップ数）、ＰＬｎ：パケットロス率、ＣＳｎ：サイト内の混雑度、ＰＳｎ：サイト内のパケット数、ＥＳｎ：パケットエラー、ＣＰＵｎｍ：ＣＰＵ負荷率、ＩＤＬＥｎｍ：ＣＰＵアイドル値、ＬＩＮＫｎｍ：コネクションリンク数、ＩＯｎｍ：ディスク負荷である。）における最適サーバ判定値Ｓｎｍが所定の条件を満たす情報配信サーバを最適な情報配信サーバとして決定させることを特徴とする。

【０１６３】請求項１５１に記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体は、分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムを制御するコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体であって、前記情報配信サーバにおいて、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせさせる問い合せステップと、前記ネットワークステータスサーバからの返答を受け取らせる返答受信ステップと、前記アクセスするクライアントに前記最適な情報配信サーバを指示させる指示ステップとを備え、前記ネットワークステータスサーバにおいて、前記問い合せステップからの問い合せを受信させる問い合せ受信ステップと、分散配置された情報配信サーバを有する各サイトとアクセスするクライアントとの間の各々の論理的な距離を取得させる論理的距離取得ステップと、前記取得させた論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定させるサイト決定ステップと、前記サイト決定ステップにより決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定させるサーバ決定ステップと、前記サーバ決定ステップにおいて決定されたサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答させる返答送信ステップとを備えることを特徴とする。

【０１６４】請求項１５２に記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体は請求項１５１に係る記憶媒体であって、更に、前記情報配信システムはルートサーバを有し、前記ルートサーバにおいて、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求めさせる論理的距離算出ステップを備え、前記論理的距離取得ステップでは、前記論理的距離算出ステップで算出させた前記論理的な距離を取得させることを特徴とする。

【０１６５】請求項１５３に記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体は請求項１５１又は１５２のいずれかに係る記憶媒体であって、更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集させるステップと、収集させた前記ネットワーク状態の情報をネットワークステータスサーバに送信させるネットワーク状態送信ステップとを備えることを特徴とする。

【０１６６】請求項１５４に記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体は請求項１５１乃至１５３のいずれかに係る記憶媒体であって、更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集させるステップと、収集させたサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信させるサイト内情報送信ステップとを備えることを特徴とする。

【０１６７】請求項１５５に記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体は、分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムを制御するコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体であって、前記情報配信サーバにおいて、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせさせる問い合せステップと、前記ネットワークステータスサーバからの返答を受け取らせる返答受信ステップと、前記アクセスするクライアントに前記最適な情報配信サーバを指示させる指示ステップとを備え、前記ネットワークステータスサーバにおいて、前記問い合せステップからの問い合せを受信させる問い合せ受信ステップと、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得させる論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集させる収集ステップと、前記論理的距離取得ステップにおいて取得させた論理的な距離と、前記収集ステップにおいて収集させた、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定させるサーバ決定ステップと、前記サーバ決定ステップにおいて決定させたサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答させる返答送信ステップとを備えることを特徴とする。

【０１６８】請求項１５６に記載のコンピュータ読み取

り可能な情報配信システム制御プログラムを格納した記憶媒体は請求項１５５に係る記憶媒体であって、更に、前記情報配信システムはルートサーバを有し、前記ルートサーバにおいて、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求めさせる論理的距離算出ステップを備え、前記論理的距離取得ステップでは、前記論理的距離算出ステップで算出させた前記論理的な距離を取得させることを特徴とする。

【０１６９】請求項１５７に記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体は請求項１５５又は１５６のいずれかに係る記憶媒体であって、更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集させるステップと、収集させた前記ネットワーク状態の情報をネットワークステータスサーバに送信させるネットワーク状態送信ステップとを備えることを特徴とする。

【０１７０】請求項１５８に記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体は請求項１５５乃至１５７のいずれかに係る記憶媒体であって、更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集させるステップと、収集したサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信させるサイト内情報送信ステップとを備えることを特徴とする。

【０１７１】請求項１５９に記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体は、分散配置された情報配信サーバと、ネットワークステータスサーバとからなる情報配信システムを制御するコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体であって、前記情報配信サーバにおいて、アクセスするクライアントに応じて、前記分散配置された情報配信サーバの中で最適な情報配信サーバを、前記ネットワークステータスサーバに問い合わせさせる問い合せステップと、前記ネットワークステータスサーバからの返答を受け取らせる返答受信ステップと、前記アクセスするクライアントに前記最適な情報配信サーバを指示させる指示ステップとを備え、前記ネットワークステータスサーバにおいて、前記問い合せステップからの問い合せを受信させる問い合せ受信ステップと、前記分散配置された情報配信サーバを有する各サイトとアクセスしてきたクライアントとの間の各々の論理的な距離を取得させる論理的距離取得ステップと、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とを収集させる収集ステップと、前記クライアントからのアクセスが最初のアクセスであると判断した場合、前記論理的距離取得ステップにおいて取得した論理的な距離にもとづいて、前記分散配置された情報配信サーバを有する各サイトから最適なサイトを決定させるサイト決定ステップと、前記サイト決定ステップにおいて決定された最適なサイト内の情報配信サーバから、所定の条件に基づいて最適な情報配信サーバを決定させる第１のサーバ決定ステップと、前記クライアントからのアクセスが最初のアクセスでないと判断した場合、前記論理的距離取得ステップにおいて取得させた論理的な距離と、前記収集ステップにおいて収集させた、前記分散配置された情報配信サーバを有する各サイトと前記アクセスしてきたクライアントとの間のネットワークの状態の情報と、前記分散配置された情報配信サーバを有するサイト内の状態の情報とに基づいて、前記分散配置された情報配信サーバから最適な情報配信サーバを決定させる第２のサーバ決定ステップと、前記第１のサーバ決定ステップもしくは前記第２のサーバ決定ステップにおいて決定させたサーバを前記最適なサーバとして、問い合わせてきた前記情報配信サーバに返答させる返答送信ステップとを備えることを特徴とする。

【０１７２】請求項１６０に記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体は請求項１５９に係る記憶媒体であって、更に、前記情報配信システムはルートサーバを備え、前記ルートサーバにおいて、サイト間の経路情報から所定のネットワークアドレスに対する論理的な距離を求める論理的距離算出ステップを備え、前記論理的距離取得ステップでは、前記論理的距離算出ステップで求めた前記論理的な距離を取得させることを特徴とする。

【０１７３】請求項１６１に記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体は請求項１５９又は１６０のいずれかに係る記憶媒体であって、更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトと前記アクセスするクライアントの間のネットワーク状態の情報を収集させるステップと、収集させた前記ネットワーク状態の情報をネットワークステータスサーバに送信させるネットワーク状態送信ステップとを備えることを特徴とする。

【０１７４】請求項１６２に記載のコンピュータ読み取り可能な情報配信システム制御プログラムを格納した記憶媒体は請求項１５９乃至１６１のいずれかに係る記憶媒体であって、更に、前記情報配信サーバにおいて、該情報配信サーバが属するサイトのネットワーク状態の情報と、該情報配信サーバの状態の情報とを収集させるステップと、収集させたサイト内の前記ネットワーク状態の情報と、該情報配信サーバの状態の情報とを、ネットワークステータスサーバに送信させるサイト内情報送信ステップとを備えることを特徴とする。

【０１７５】

【発明の実施の形態】以下、本発明の実施形態について図面を参照して詳細に説明する。図１は本発明の最適サーバ選択システムの構成システム図である。各サイト間の通信手段としてインターネットを使用した例を示している。図１において、１は情報配信するサーバにアクセスする端末側のサイト、２はアクセスされる情報配信するサーバ（以下、情報配信サーバもしくはｗｅｂサーバとも称する）を有するサイト（サーバサイト）である。ここで、サイトとは、同一の論理的なネットワークブロックを意味しており、例えば、ネットワークアドレスクラスで言うところのクラスＢあるいは、クラスＣのアドレスブロックなどの事である。もちろん、物理的に同一の場所にあるときにもサイトとして見る事が可能である。また、本実施形態では、サーバサイト２は、高速大容量の情報配信を行うために、分散して配置されている。図１では、分散配置されたサーバサイト１２として示している。３は各サイト間の通信手段であるインターネットである。本実施形態では、各サイト間の通信手段をインターネットとしているが、これに限る事無く、例えば、パケット通信網、公衆網（電話網）などでも良い。

【０１７６】４は、情報配信サーバにアクセスする端末であり、本実施形態では、情報配信サーバ（ｗｅｂサーバ）にアクセスできる標準ブラウザーがインストールされている情報処理装置とする。５はアクセス側のサイト内と外との経路制御を行うためのルータである。６は情報配信サーバがあるサイト内と外との経路制御を行うためのルータである。

【０１７７】７は情報配信サーバで、本実施形態では、標準的な情報配信用ソフトであるｗｅｂサーバソフト８と本発明で新たに作成したａｇｅｎｔソフト９が動作している。ｗｅｂサーバソフト８は、ｈｔｍｌ言語で書かれた情報などを外部からのリクエストに従って転送する機能を持つサーバソフトである。また、ａｇｅｎｔソフト９は、サーバの種々な動作状況やｗｅｂサーバソフト８に対しての指示などを受け持つソフトである。尚、情報配信サーバは、ｗｅｂサーバソフト８とａｇｅｎｔソフト９が一体となったソフトウェアで動作していてもよいことは言うまでもない。

【０１７８】各サイトのルータ６は、各サイトからインターネット（ネットワーク）を介して他のサイトを見たときの経路情報をそれぞれ有している。ルートサーバ１０は、地域的、論理的に分散している情報配信サーバ内の、各々のサイト側ルータ６から経路情報（例えば、ＢＧＰ情報、ＲＩＰ情報、ＯＳＰＦ情報など）を取得する機能を持つ。また、ルートサーバ１０は、ネットワークステータスサーバ（以後ＮＳサーバとする）１１に対して経路情報を送信する機能を持つ。

【０１７９】ＮＳサーバ１１は、分散配置された情報配信サーバを有するサイト２やサイト１２内の情報配信サーバの各種動作状態をａｇｅｎｔ９に対して測定指示し、その測定結果を取得、蓄積する。またａｇｅｎｔ９は情報のリクエストをしてきた端末側サイト１と各情報配信サーバサイト２や１２との間のネットワーク的な状態（例えば、混雑度、ルータ段数、パケットロス率、応答時間など）を計測し、ＮＳサーバ１１はａｇｅｎｔ９が計測したネットワーク的な状態を取得、蓄積する。さらにＮＳサーバ１１は、情報のリクエストをしてきた端末側サイト１と各情報配信サーバを有するサイト２や１２とのネットワーク的な距離を求めるためにルートサーバ１０から情報を取得する。

【０１８０】尚、図１では、ルートサーバ１０及びＮＳサーバ１１は、サイト２内にあるが、サイト２に限るものではなく、それぞれ他のどのサイトにあってもよい。

【０１８１】次に、本実施形態の最適なｗｅｂサーバへのアクセス制御動作について説明する。まず図２は、ｗｅｂサーバアクセスを模式的に示した図である。ここでは、サイト１の端末４がｗｅｂサーバにアクセスした時の具体的な動作を示している。ＡＳ（Autonomous system,自律システム）１内の端末４が、www.abcde.jpなるｗｅｂサイトのアクセスアドレス（以降ＵＲＬとする）を持つｗｅｂサーバにアクセスした場合について示している。最初に、ＵＲＬから目的のｗｅｂサーバのＩＰアドレスを求めるために、自ネットワーク（サイト１）内の最寄りのＤＮＳ(Domain Name Server)に問い合わせ、対応するＩＰアドレス（addr1）を受け取る（（1））。そして、ＩＰアドレス（addr1）によって、所望のｗｅｂサイト（www.abcde.jpなるＵＲＬを持つｗｅｂサイト）にアクセス（＝リクエスト）する（(2)）。ＩＰアドレス（addr1）を持つｗｅｂサーバAでは、アクセスを受けつけると、ａｇｅｎｔ９によって、アクセスしてきたクライアントに対して自ｗｅｂサーバが最適サーバか否かをＮＳサーバ（ネットワークステータスサーバ）に問い合わせる（（3））。ＮＳサーバでは、分散配置されているｗｅｂサーバを有する各サイト２，３，４からアクセスしてきたクライアントサイト１までのネットワーク的な距離をルートサーバ１０に問い合わせる（（４））。ルートサーバ１０では、各サイト内のボーダーゲートウェイルータ６と経路情報の交換を行っているため、即座に各サイトからクライアントサイト１までの距離を返答が可能である（（５））。ＮＳサーバ１１は、ルートサーバ１０からの返答を受けて、ネットワーク的に近いサイトであるサイト３を決定し、サイト４のａｇｅｎｔ９に最適ｗｅｂサーバＢのURLを返答する（（６））。または、最適ｗｅｂサーバのIPアドレス（addr2）を返答しても良い。本実施形態では、この時点（クライアントが最初にアクセスしてきたとき）のサイトの選択方法として、即座に情報を取得できる経路情報のＢＧＰ（Border Gateway Protocol）のＡＳパス情報を用いる。そのため各ｗｅｂサイトからクライアント

サイトまでのＡＳパスの短いサイトであるサイト３を選択し、さらに所定の方法を用いて該サイト３内のaddr2をもつｗｅｂサーバＢを最適ｗｅｂサーバとして選択している。

【０１８２】ただし、ＮＳサーバでは、この経路情報の他にも、各サイトからクライアントまでの間にあるルータの数（ルータホップ数）、応答時間、パケットロス率などのネットワーク情報や、各サイト内のネットワーク情報（混雑度、パケット数、パケットエラーなど）、サーバの負荷状況（ＣＰＵ負荷率、ＣＰＵアイドル率、コネクションリンク数、ディスク負荷など）などの情報を取得するよう動作する。ＮＳサーバは、次回の問い合わせに備えてその情報を蓄積（一時蓄積も含む）しておく（（7））。

【０１８３】ｗｅｂサーバＡ内のａｇｅｎｔ９は、次のアクセスの際にはｗｅｂサーバＢにアクセスするようにクライアント４に指示する（（8））。例えば、ＨＴＭＬ言語のロケーション命令を使って自動的にクライアントが接続するｗｅｂサーバを変更させることができる。その結果、クライアント４は、次のアクセスからは、自動的に最適ｗｅｂサーバであるＢにアクセスをする事になる。

【０１８４】次に、クライアント４が引き続き、ｗｅｂサーバＢにアクセスしている時の動作を同じく図２を用いて説明する。前述の過程を経てｗｅｂサーバＢにアクセスをする（（９））。ｗｅｂサーバＢでは、アクセスを受けつけると、ａｇｅｎｔ９によって、アクセスしてきたクライアントに対して自ｗｅｂサーバが最適サーバか否かをＮＳサーバ１１に問い合わせる（（10））。ＮＳサーバ１１では、上述した最初のアクセス時の動作と同じく、経路情報の収集及び各種状態情報を総合して最適サーバサイトと最適ｗｅｂサーバを決定する。ＮＳサーバ１１では、前回のアクセスによって取得、蓄積されている経路情報に加えて、前回のアクセス時には調査ができていなかった、各サイトからクライアントまでの間にあるルータの数（ルータホップ数）、応答時間、パケットロス率などのネットワーク情報や、各サイト内のネットワーク情報（混雑度、パケット数、パケットエラーなど）、サーバの状態の情報（ＣＰＵ負荷率、ＣＰＵアイドル率、コネクションリンク数、ディスク負荷など）などを取得しているので、その取得した各種情報から、クライアントがアクセスすべき最適なｗｅｂサーバを計算、決定する。この時点での選択方法としては、各情報に対し、その内容に応じて数値的に重みを課して、その重みの合計を各サイト、各サーバ毎に比較する事によって最適なｗｅｂサーバＣを決定する（（１１））。ＮＳサーバ１１は、ｗｅｂサーバＢ内のａｇｅｎｔ９に通知し（（１２））、さらに、その旨をクライアント４に返答する（（13））。結果、クライアント４は、次のアクセスからは、自動的に最適ｗｅｂサーバであるＣにアクセスをする事になる（（１４））。

【０１８５】さらに、クライアント４が引き続き、ｗｅｂサーバＣにアクセスした場合には、上述したｗｅｂサーバＢへのアクセスと同様な動作をするが（（１５）、（１６））、このＮＳサーバへのアクセス動作は、所定時間後に行うものとする。そして、所定時間内に、取得、蓄積された各種ネットワーク状態情報、サーバ状態情報によって新たな最適ｗｅｂサーバが決定された時には、クライアント４はその新たなｗｅｂサーバにアクセスを変更する事になる。

【０１８６】次に、本実施形態の最適ｗｅｂサーバへのアクセス制御動作について図３のフローチャートをもとにその動作を説明する。図３－１は、アクセスクライアントが最初にＷｅｂサーバＡにアクセスしてから、最初の最適ｗｅｂサーバＢへアクセスを変更するまでの動作フローチャートである。図３－２は、その時のＮＳサーバの動作フローチャートを示した図である。図３－３は、次にアクセスクライアントがｗｅｂサーバＢにアクセスしてから、最適アクセスｗｅｂサーバＣへアクセスを変更するまでの動作フローチャートである。図３－４は、その時のＮＳサーバの動作フローチャートを示した図である。そして、図３－５は、アクセスクライアントがｗｅｂサーバＣにアクセスしてから、さらに最適アクセスｗｅｂサーバを調査した上で、再度ｗｅｂサーバＣへアクセスをするまでの動作フローチャートである。

【０１８７】最初に、図３－１、図３－２をもとに説明する。ＵＲＬから目的のｗｅｂサーバのＩＰアドレスを求めるために、自ネットワーク内（サイト１）の最寄りのDNS(Domain Name Server)に問い合わせ、対応するＩＰアドレス（addr1）を受け取る（Ｓ１）。

【０１８８】そして、ＩＰアドレス（addr1）によって、所望のｗｅｂサイトＡにアクセスする（Ｓ２）。

【０１８９】ｗｅｂサーバＡでは、アクセスを受けつけると、ａｇｅｎｔ９によって、アクセスしてきたクライアントに対して自ｗｅｂサーバが最適サーバか否かをＮＳサーバに問い合わせる（Ｓ３）。

【０１９０】ＮＳサーバでは、分散配置されている各ｗｅｂサーバサイトからアクセスしてきたクライアントサイト１までのネットワーク的な距離をルートサーバ１０に問い合わせる。（Ｓ４）。

【０１９１】ルートサーバ１０では、各サイト内のボーダーゲートウェイルータ６と経路情報の交換を行っているため、即座に各サイトからクライアントサイト１までの距離を返答することが可能であり、ＮＳサーバはその経路情報と距離を取得する（Ｓ５）。

【０１９２】ＮＳサーバ１１は、ルートサーバ１０からの返答を受けて、ネットワーク的に近いサイトであるサイト３を決定し、所定の方法でそのサイト３内の最適ｗｅｂサーバＢを決定して、サイト４のａｇｅｎｔ９に、最適なｗｅｂサーバがｗｅｂサーバＢである旨を返答す

る（Ｓ６）。尚、Ｓ６で最適ｗｅｂサーバを決定する際の所定の方法とは、予めサイト毎に決めておいたｗｅｂサーバでもよいし、サイト毎に所定の順序でｗｅｂサーバを決めるようにしてもよいし、任意にそのサイト内のｗｅｂサーバを決めるようにしてもよいし、また、ＮＳサーバはｗｅｂサーバを有するサイト内の状態の情報（サイト内のネットワーク状態の情報（混雑度、パケット数、パケットエラーなど）、サーバの状態の情報（ＣＰＵ負荷率、ＣＰＵアイドル率、コネクションリンク数、ディスク負荷など））を取得しているので、そのサイト内の状態の情報を利用して最適ｗｅｂサーバを決定するようにしてもよい。

【０１９３】ＮＳサーバは、次回の問い合わせに備えてその取得した経路情報を蓄積（一時蓄積も含む）しておく（Ｓ７）。

【０１９４】一方、ｗｅｂサーバA内のａｇｅｎｔ９は、次のアクセスからは最適ｗｅｂサーバBにアクセスするようにクライアント４に返答する（Ｓ８）。

【０１９５】そして結果、クライアント４は、次のアクセスからは、自動的に最適ｗｅｂサーバBにアクセスをする事になる。

【０１９６】次に、図３－３、図３－４をもとに説明する。

【０１９７】前述の過程を経てｗｅｂサーバBにアクセスをする（Ｓ９）。

【０１９８】ｗｅｂサーバBでは、アクセスを受けつけると、ａｇｅｎｔ９によって、アクセスしてきたクライアントに対して自ｗｅｂサーバが最適サーバか否かをＮＳサーバ１１に問い合わせる（Ｓ１０）。

【０１９９】ＮＳサーバ１１では、上述した最初のアクセス時の動作と同じく、経路情報の収集及び各種状態情報を総合して最適サーバサイトと最適ｗｅｂサーバを決定する。ＮＳサーバ１１では、前回のアクセスによって取得、蓄積されている経路情報に加えて、前回のアクセス時には調査ができていなかった各サイトからクライアントまでの間にあるルータの数（ルータホップ数）、応答時間、パケットロス率などのネットワーク情報に加え、各サイト内のネットワーク状態の情報（混雑度、パケット数、パケットエラーなど）、各ｗｅｂサーバの状態の情報（ＣＰＵ負荷率、ＣＰＵアイドル率、コネクションリンク数、ディスク負荷など）などを取得しているので、その取得した各種情報から、クライアントがアクセスすべき最適なｗｅｂサーバを計算、決定する。この時点での選択方法としては、各情報に対し、その内容に応じて数値的に重みを課して、その重みの合計を各サイト、各サーバ毎に比較する事によって最適なｗｅｂサーバCを決定する（Ｓ１１）。

【０２００】ＮＳサーバ１１は、ｗｅｂサーバB内のａｇｅｎｔ９に最適なｗｅｂサーバCを通知し（Ｓ１２）、さらに、次のアクセスからは最適ｗｅｂサーバCにアクセスするようにクライアント４に返答する（Ｓ１３）。

【０２０１】結果、クライアント４は、次のアクセスからは、自動的に最適ｗｅｂサーバCにアクセスをする事になる（Ｓ１４）。

【０２０２】最後に図３－５を用いて説明する。クライアント４が引き続き、ｗｅｂサーバCにアクセスした場合には、上述したｗｅｂサーバBへのアクセスと同様に、ＮＳサーバに最適ｗｅｂサーバを問い合せるが（Ｓ１５）、ここでは、前回と同様にｗｅｂサーバCが最適Ｗｅｂサーバと判定されたため、アクセスするｗｅｂサーバの変更処理（Ｓ１６）は行われない。ただし、Ｓ１５でｗｅｂサーバCと異なるｗｅｂサーバが最適なｗｅｂサーバとして返答されたときは、アクセスするｗｅｂサーバの変更処理を行なうことは言うまでもない。

【０２０３】尚、このＮＳサーバへのアクセス動作は、所定時間後に間欠的に行う。そして、所定時間内に、取得、蓄積された各種ネットワーク状態情報、サーバ状態情報によって新たな最適ｗｅｂサーバが決定された時には、クライアント４はその新たなｗｅｂサーバにアクセスを変更することになる。

【０２０４】次に図４を用いてＮＳサーバ１１が、ｗｅｂサーバからクライアントまでのネットワークの状態やｗｅｂサーバのサイト内のネットワークの状態やｗｅｂサーバの状態に関する情報の取得、蓄積する際の動作について説明する。

【０２０５】本方式では、下記の示すような２つのカテゴリーの情報を定期的（間欠的）に収集する。

（１）各サーバのサイトからクライアントまでのネットワーク状態

（２）各サーバのサイト内の状態（ネットワーク状態、サーバの状態）

図４では、その関係を図に示している。ＮＳサーバ１１は、各サーバサイトからアクセスしたクライアントまでのネットワーク状態を定期的に計測する。計測する項目を次項にまとめておいた。

【０２０６】各サーバサイトのルータ６は、該サーバサイトから見たネットワークの経路情報を保持しており、ルートサーバ３は各サーバサイトのルータ６から該ルータの有する経路情報を収集している。ルートサーバ３は、各サーバサイトのルータ６：ＢＧＲ（Border gateway router:外部ネットワーク境界ルータ）と論理的に接続（論理的にpeering）している（①）。そして、ＮＳサーバ１１からのリクエストに応じて所定の経路情報の分析結果を求め、ＮＳサーバ１１に返答する（③）。他の調査項目については、ＮＳサーバが自律的に実施するが（②）、調査内容に応じて、各ｗｅｂサーバソフトウェアに組み込んでいるａｇｅｎｔソフトウェア９に、ＮＳサーバ１１が指示をして、調査する場合もある。

【０２０７】各サーバサイト内のネットワーク状態やサ

ーバ状態については、各サーバサイトのｗｅｂサーバソフトに組み込んでいるａｇｅｎｔソフトウェア９が、ＮＳサーバ１１からの指示に従って定期的に調査する。収集された種々の情報は、時系列的にＮＳサーバに蓄積される。このデータ蓄積方法は、間欠的に種々の情報について時間を付属情報として蓄積する。本実施形態では、各種情報を３０分毎に調査、取得、蓄積をする。また、蓄積された情報は、日単位、週単位、月単位、年単位などの所定の期間で周期的に変化する事が経験的に知られている。そこで、本方式では、蓄積した情報を所定の関数で近似する事を行う。

【０２０８】近似する時間単位は、日単位、週単位、月単位、年単位のいずれであってもよいが、本実施形態では、各種情報を日単位で３次関数により近似し係数を保存する。例えば、サーバサイト内のネットワークの混雑度（ＣＳｎ）を３次関数で近似した場合、時間軸をｔとして、図8のような近似曲線で近似されるものとする。この近似式がＣＳｎ＝ａ・ｔ＾３＋ｂ・ｔ＾２＋ｃ・ｔ＋ｄ、（ただし、ａ、ｂ、ｃ、ｄ：係数）という式で表された場合、ＮＳサーバはこの係数（３次関数自体でもよい）を保存する。

【０２０９】この結果、各サーバから最適サーバの選択リクエストを受けた時には、この３次関数を利用することにより、瞬時に各種状態を推定し、その推定値をもとに最適サーバの選択を行う。

【０２１０】各サーバサイトとアクセスクライアントとの間のネットワークの状態情報の収集は、無限に続けるわけでなく、分散配置されたサーバにアクセスを継続している間、及び、その後所定期間続ける。本実施形態では、1週間の期限を設けることにより、アクセスクライアント４から１週間アクセスが無い場合には、そのアクセスクライアント４への各サイトからの情報収集動作を実施しない。

【０２１１】また、アクセスクライアント４から所定の期間（例えば３週間）、アクセスがない場合は、ＮＳサーバ内に蓄積されたそのアクセスクライアント４と各サーバサイトとの間のネットワークの状態情報（近似式など）を、破棄するようにしてもよい。そうすれば、無駄な情報を蓄積することがなくなり、ＮＳサーバの必要とする記憶容量を節約することもできるという効果もある。

【０２１２】さらに、蓄積する情報は、特定エリアを代表した形での結果として扱う事も可能である。つまり、ＮＳサーバは、各サーバサイトとアクセスクライアントとの間の経路情報（論理的な距離）や各サーバサイトとアクセスクライアントとの間のネットワークの状態情報を取得して蓄積するが、各サーバサイトとアクセスクライアントとの間の経路情報（論理的な距離）は、各サーバサイトとアクセスクライアントの属するサイトとの間の経路情報(論理的な距離)を利用してもよいし、各サーバサイトとアクセスクライアントとの間のネットワークの状態情報は、各サーバサイトとアクセスクライアントの属するサイトとの間のネットワークの状態情報を利用してもよい。

【０２１３】例えば、アクセスクライアント４のＩＰアドレスが、192.168.0.1だとすると、このクライアントとのネットワーク状態をクラスＣアドレスブロック全体の結果とする。具体的には、192.168.0.0/24のネットワークブロックからのアクセスに対しては、既に調査し取得している192.168.0.1の情報を使用する。こうする事によって、蓄積すべきデータに必要な記憶スペースを少なくする事ができる。本実施形態では、クラスＢアドレス（A.B.0.0/16に相当するネットワークアドレスブロック）でアクセスクライアントエリアを分類している。

【０２１４】次に、ＮＳサーバ１１が自立的に調査・取得するネットワーク及び情報配信サーバの状態の情報にどのようなものがあるか述べる。

（１）各情報配信サーバサイトからクライアントまでのネットワーク状態の情報には以下のようなものがある。

・ＡＳＬｎ：ASパスによる論理的なネットワーク間距離（BGPの経路情報を用いる）
・ＲＴｎ：応答時間
・ＲＮｎ：ルータ段数（ルータホップ数）
・ＰＬｎ：パケットロス率

（２）各情報配信サーバサイト内の状態（ネットワーク、サーバ）の情報には以下のようなものがある。

【０２１５】サイト内のネットワーク状態の情報として、

・ＣＳｎ：サイト内の混雑度
・ＰＳｎ：サイト内のパケット数
・ＥＳｎ：パケットエラー

情報配信サーバの状態情報として、

・ＣＰＵｎｍ：ＣＰＵ負荷率
・ＩＤＬＥｎｍ：ＣＰＵアイドル値
・ＬＩＮＫｎｍ：コネクションリンク数
・ＩＯｎｍ：ディスク負荷

但し、ｎ：サーバサイト番号、ｍ：サーバ番号

これらのネットワーク及びサーバの状態の情報を用いて最適サーバを決定する際の方法について述べる。ＮＳサーバ１１では、これら各種の状態情報が蓄積されており、これらの情報を用いて最適なサーバサイト、及びサーバを決定する。本実施形態では、以下の式を用いて計算で求める。

【０２１６】選択方法１は、図３－２におけるステップＳ６に対応する選択方法であり、ＡＳＬｎが最小のサイトを最適サイトとする。

【０２１７】選択方法２は、図３－４におけるステップＳ１１に対応する選択方法である。

【０２１８】各サーバーサイト毎に下記の式で示すように該サーバサイトとクライアント間のネットワーク状態

値：Ｋ１ｎ、サーバサイト内ネットワーク状態値：Ｋ２ｎ、サーバ状態値：Ｋ３ｎｍを求め、最適サイト判定値Ｋｎを求める。この値が最小であるサイト番号（ｎ）が最適サイトとする。そして、Ｋ３ｎｍを用いて最適サーバ判定値：Ｓｎｍを求める。この値は、最適サイト判定値とサーバ状態値から最適サーバ判定値を求めている。そのため、クライアントのサイトとサーバサイトとのネットワーク状態、サイト内のネットワーク状態、サーバ状態、及び論理的なネットワーク間の距離をＡ～Ｏによる重み係数を用いる事によって、最適サーバを総合的に判断をする。なお、ここでは、このＳｎｍが最小であるサーバが最適サーバとして決定することにする。

【０２１９】

ネットワーク状態値：Ｋ１ｎ＝ＲＴｎ・Ａ＋ＲＮｎ・Ｂ＋ＰＬｎ・Ｃ

サイト内ネットワーク状態値：Ｋ２ｎ＝ＣＳｎ・Ｄ＋ＰＳｎ・Ｅ＋ＥＳｎ・Ｆ

サーバ状態値：Ｋ３ｎｍ＝ＣＰＵｎｍ・Ｇ＋ＩＤＬＥｎｍ・Ｈ＋ＬＩＮＫｎｍ・Ｉ＋ＩＯｎｍ・Ｊ

最適サイト判定値：Ｋｎ＝Ｋ１ｎ・Ｋ＋Ｋ２ｎ・Ｌ＋ＡＳＬｎ・Ｍ

最適サーバ判定値：Ｓｎｍ＝Ｋｎ・Ｎ＋Ｋ３ｎｍ・Ｏ

但し、ｎ：サーバサイト番号、ｍ：サーバ番号、Ａ～Ｏ：重み係数である。

【０２２０】また、本実施形態では、ネットワークの状態値として、応答時間、ルータ段数（ルータホップ数）、パケットロス率を用い、サイト内のネットワーク状態値として、サイト内の混雑度、サイト内のパケット数、パケットエラーを用い、サーバ状態値として、ＣＰＵ負荷率、ＣＰＵアイドル値、コネクションリンク数、ディスク負荷を用いたが、ネットワーク状態値及びサイト内ネットワーク状態値及びサーバ状態値は、本実施形態でそれぞれが利用した情報の少なくとも１つを利用するようにしてもよい。

【０２２１】また、サーバサイト内の状態の情報として、サイト内のネットワークの状態情報と情報配信サーバの状態情報とを利用したが、どちらか一つを代表させて最適サーバを決定する際に使用するようにしてもよい。

【０２２２】次に、図５を用いて本方式で用いたルートサーバの機能について述べる。現在のインターネット／イントラネットは、ルータがメッシュ状に接続された環境であると言える。その中で、所望のパケットを所定の場所に配送するために、ルーティングプロトコルなるものが使われており、特に、広域ネットワークにおいては、Border Gateway Protocol(BGP)というルーティングプロトコルを使用している。BGPでは、各ルータに経路テーブルが用意されており、この経路テーブルを基にパケットを所定の場所に配送している。

【０２２３】図５では、ルートサーバ１０と各サイトのルータとＢＧＰによって論理的にpeeringされている。ここでは、各サイトとクライアントとの距離を計測する一手法として、AS path数を利用している。このAS-path数とは、BGPの中で扱われている独立したネットワーク数の事で、経路情報の中には、所望のAS番号までの経路情報がAS番号の繋がりとしてマッピングされている。つまり、サーバサイトとクライアントサイトの間の経路情報がAS番号という独立したネットワークに与えられている番号の繋がりとして知る事ができ、AS-path数は、独立したネットワークを経由する回数を示すもので、このAS番号の繋がりから知ることができる。

【０２２４】ルートサーバは、この経路情報を各サーバーサイトにあるルータ６から取得することによって、各サーバサイトから見た経路情報を集約できるため、アクセスクライアントに（AS path的に）最も近いサーバサイトを選ぶ事ができる。

【０２２５】本実施形態では、このルートサーバを用いて、アクセスクライアントと各サーバサイトとのＢＧＰ的な距離、すなわちAS-PATH数を求め、その距離を最適サーバサイト決定のための選択条件の一つとして使用している。

【０２２６】また、本実施形態では、選択方法１として、サーバサイトからクライアントまでの論理的な距離から最適サイトを求めた後、所定の方法で該最適サイト内の情報配信サーバを決定する方法と、選択方法２として、論理的な距離及びネットワークの状態情報及び情報配信サーバの状態情報から、最適な情報配信サーバを決定する方法とを用い、クライアントが最初にアクセスしてきたときは選択方法１を用いて最適サーバを決定し、２回目以降にアクセスしてきたときは選択方法２を用いて最適サーバを決定したが、本発明はこの順序に限るものではなく、常に選択方法１を使用するようにしてもよいし、常に選択方法２を使用するようにしてもよい。

【０２２７】また、ＮＳサーバは、クライアントがアクセスしてきたときに、該クライアントとサーバサイトとの間の経路情報（論理的な距離）が蓄積されていた場合、蓄積されている経路情報を利用して最適サーバを決定するようにしてもよい。

【０２２８】また、本発明に係るＮＳサーバや情報配信サーバやルートサーバは、図９に示すようなコンピュータと同等の構成をもつ情報処理装置によって実現されうる。図９のコンピュータ９０１は、プログラムを読み出して実際の処理を行なうＣＰＵ９０２、処理を行なう際にＣＰＵ９０２が使用するＲＡＭ９０３、プログラムコードを供給する記憶媒体であるＨＤ（ハードディスク）９０４、ＦＤ（フロッピー（登録商標）ディスク）９０５、ネットワークと接続する際に使用する通信インタフェース９０６などから構成される。本発明に係るネットワークステータスサーバ制御プログラムや、情報配信システム制御プログラムは、外部からインストールされる

プログラムによって、図９と同等の構成を持つコンピュータによって遂行されてもよい。その場合、図１０のように、前記実施形態の機能を実現するソフトウェアのプログラムコードを記録した１００２のような記録媒体をシステムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはＣＰＵやＭＰＵ）が記憶媒体に格納されたプログラムコードを読み出し実行することによっても、本発明の目的が達成されることは言うまでもない。

【０２２９】この場合、記憶媒体から読み出されたプログラムコード事体が本発明の新規な機能を実現することになり、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。

【０２３０】プログラムコードを供給するための記憶媒体としては、例えば、フロッピーディスク、ハードディスクなどの磁気ディスクや、光ディスク、光磁気ディスク、ＣＤ－ＲＯＭ、ＣＤ－Ｒ、ＤＶＤ－ＲＯＭ、ＤＶＤ－ＲＡＭ、磁気テープ、メモリカード、ＲＯＭなどを用いることができる。

【０２３１】また、本発明は、前記実施形態の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体から、そのプログラムをパソコン通信など通信ラインを介して配信する場合にも適用できることは言うまでもない。

【０２３２】図１１は、プログラムコードを供給するＣＤ－ＲＯＭなどの記憶媒体のメモリマップの一例を示す図である。１１０１はディレクトリ情報を記憶している領域で、インストールプログラムを記憶している領域１１０２、及びネットワークステータスサーバ制御プログラムもしくは情報配信システム制御プログラムなどの制御プログラム１１０３を記憶している領域の位置を示している。本発明のネットワークステータスサーバ制御プログラムもしくは情報配信システム制御プログラムがコンピュータにインストールされる際には、まず領域１１０２に記憶しているインストールプログラムがシステムにロードされ、ＣＰＵによって実行される。次に、ＣＰＵによって実行されるインストールプログラムが、ネットワークステータスサーバ制御プログラムもしくは情報配信システム制御プログラムなどの制御プログラムを記憶している領域１１０３からネットワークステータスサーバ制御プログラムもしくは情報配信システム制御プログラムを読み出して、ハードディスクに格納する。

【０２３３】本発明を上記記憶媒体に適用する場合、その記憶媒体には、先に説明したフローチャートに対応するプログラムコードを格納することになる。

【０２３４】また、本発明の目的は、前述した実施形態における様々な機能を実現するソフトウェアのプログラムコードを、図１０に示すように、送出装置１００４、例えばＨＴＴＰサーバやＦＴＰサーバなどから送出してもらい、それを１００５に示されるネットワーク又は公衆回線又は無線などを介して受け取り、そのシステム或は装置のコンピュータ（又はＣＰＵ，ＭＰＵ）がそのプログラムコードを実行することによっても達成されることは言うまでもない。

【０２３５】この場合、送出装置から送出されたプログラムコード自体が、前述した実施形態における機能を実現することになり、よって、そのプログラムコードを送出する送出装置は本発明を構成することになる。

【０２３６】また、コンピュータが読み出したプログラムコードを実行することによって、前記実施形態の機能が実現される以外にも、そのプログラムコードの指示に基づき、コンピュータ上で稼動しているＯＳなどが実際の処理の一部または全部を行い、その処理によっても前記実施形態の機能が実現される場合にも本発明は適用できる。

【０２３７】以上説明したように、本実施形態によれば、分散配置されたｗｅｂサーバサイト間のアクセス負荷を平準化することができるという効果がある。

【０２３８】また、分散配置されたｗｅｂサーバサイト間のアクセスを平準化するとともに、各々のｗｅｂサーバサイト内のｗｅｂサーバ間においてもアクセス負荷の平準化をすることができるという効果がある。

【０２３９】また、最適なｗｅｂサーバを高速に決定するために、最初にクライアントがアクセスしてきたときは、第一の決定手法でまず最適なｗｅｂサーバサイトを経路情報（論理的な距離）を用いて高速に決定し、クライアントが次回以降にアクセスしてきたときは、第二の決定手法で、経路情報に加えてネットワークの状態の情報及びｗｅｂサーバサイト内の状態の情報を用いて最適なｗｅｂサーバを決定することで、高速に最適なｗｅｂサーバを決定することができるため、ｗｅｂサーバが高速にアクセスクライアントに応答することができるという効果がある。

【０２４０】また、最適なｗｅｂサーバを知り得るために必要なネットワーク状態情報、サーバ状態情報が、ＮＳサーバ内で自動的に取得、蓄積されており、また、ＮＳサーバ内で取得、蓄積された情報で、総合的に最適なｗｅｂサーバを決定する事により、アクセスされたｗｅｂサーバが高速に最適なｗｅｂサーバを決定することができるという効果がある。

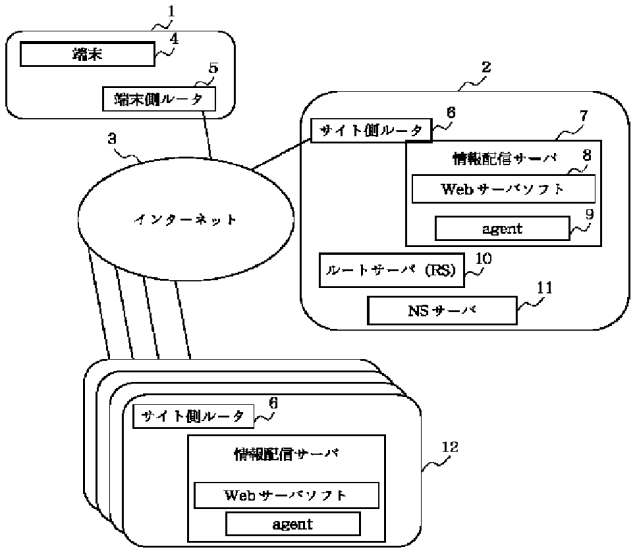
【０２４１】また、クライアントは別途付加的な装置を加える事無く、また、クライアントが何ら能動的な操作の必要も無く最適ｗｅｂサーバに切り替えられ、さらに、分散配置されたｗｅｂサーバサイト内のすべてのｗｅｂサーバによってアクセスを受け付けることができるので、アクセスの集中をなくすことができるという効果がある。

【図面の簡単な説明】

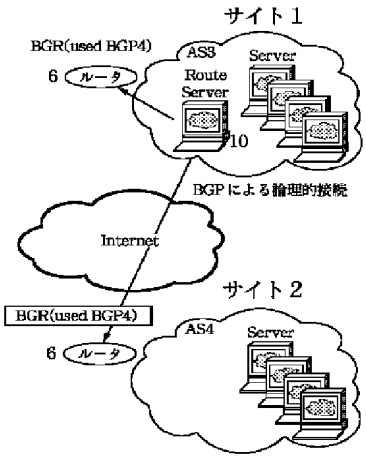
【図１】本発明の実施形態に係る構成を説明する図である。

【図２】本発明の実施形態に係る機能・動作を説明する図である。
【図3】本発明の実施形態に係る機能・動作を説明するフローチャートである。
【図４】本発明の実施形態に係るＮＳサーバの機能・動作を説明する図である。
【図５】本発明の実施形態に係るルートサーバの機能・役割を説明する図である。
【図６】従来例であるバーチャルホスト方式を説明するための図である。
【図７】従来例であるＴＣＰコネクションホップ方式を説明するための図である。
【図８】収集した情報を所定の関数で近似したときの近似曲線を示す図である。
【図９】本発明に係るＮＳサーバや情報配信サーバやルートサーバを実現しうるコンピュータの構成図である。
【図１０】プログラムコードを記憶した記憶媒体、及びプログラムコードを送出する送出装置を示す模式図である。
【図１１】本発明におけるネットワークステータスサーバ制御プログラムもしくは情報配信システム制御プログラムが記憶された記憶媒体のメモリマップを示すイメージ図
【符号の説明】
１　アクセスサイト
２　ｗｅｂサーバサイト１
３　インターネット
４　アクセス端末（アクセスクライアント）
５　端末側ルータ
６　サイト側ルータ
７　ｗｅｂサーバ（情報配信サーバ）
８　ｗｅｂサーバソフト
９　ａｇｅｎｔ
１０　ルートサーバ
１１　ＮＳサーバ
１２　ｗｅｂサーバサイト２
９０１　コンピュータ
９０２　ＣＰＵ
９０３　ＲＡＭ
９０４　ＨＤ
９０５　ＦＤ
９０６　通信インタフェース
９０７　モニタ
９０８　マウス
９０９　キーボード
１００１　コンピュータ
１００２　記憶媒体
１００３　記憶媒体読み出し及び書き込み部
１００４　送出装置
１００５　通信ライン

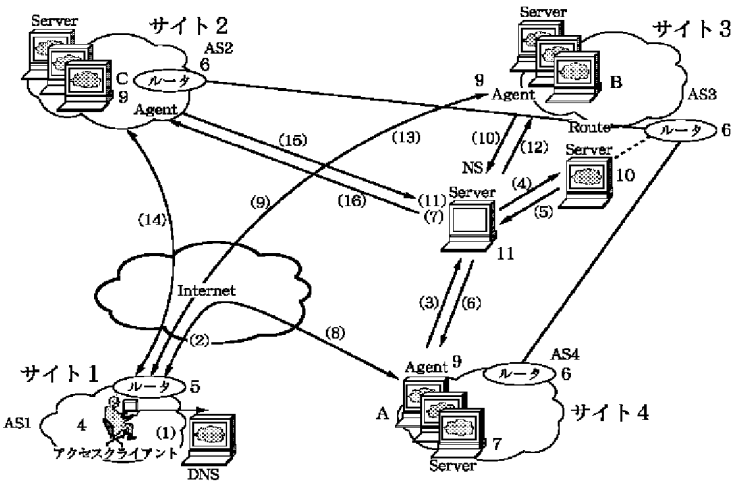
【図１】



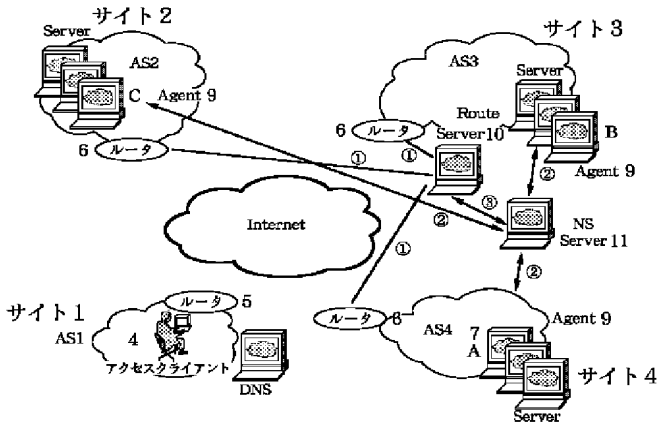
【図５】



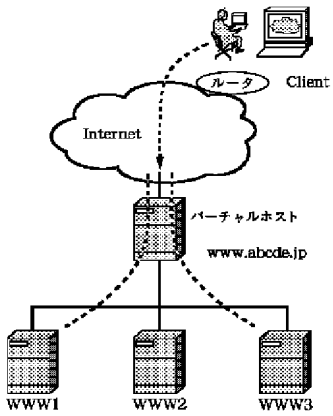
【図２】



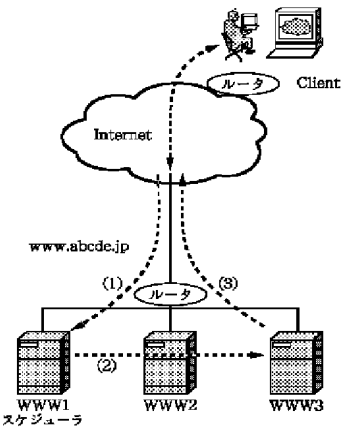
【図４】



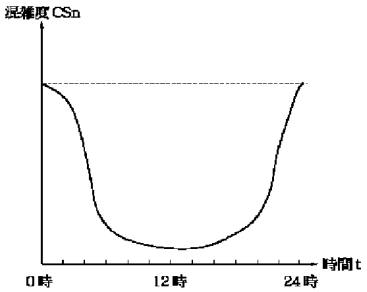
【図６】



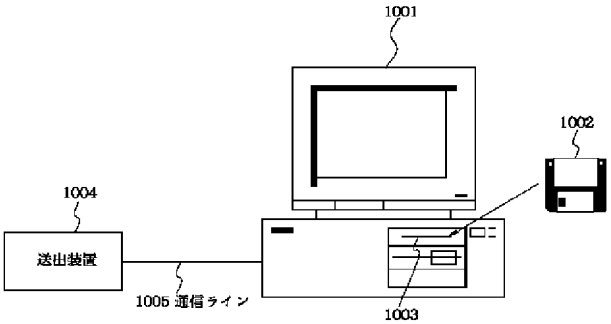
【図７】



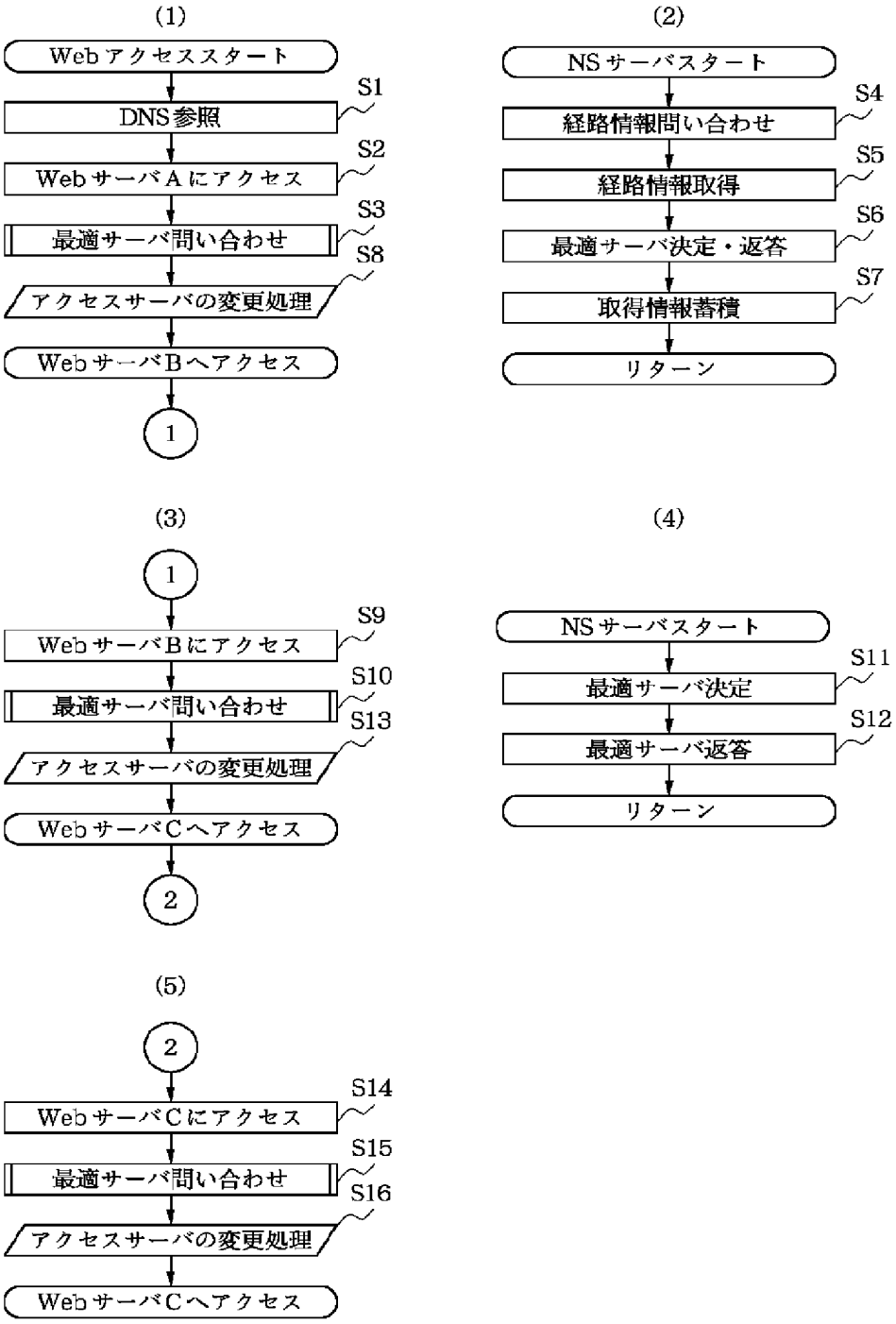
【図８】



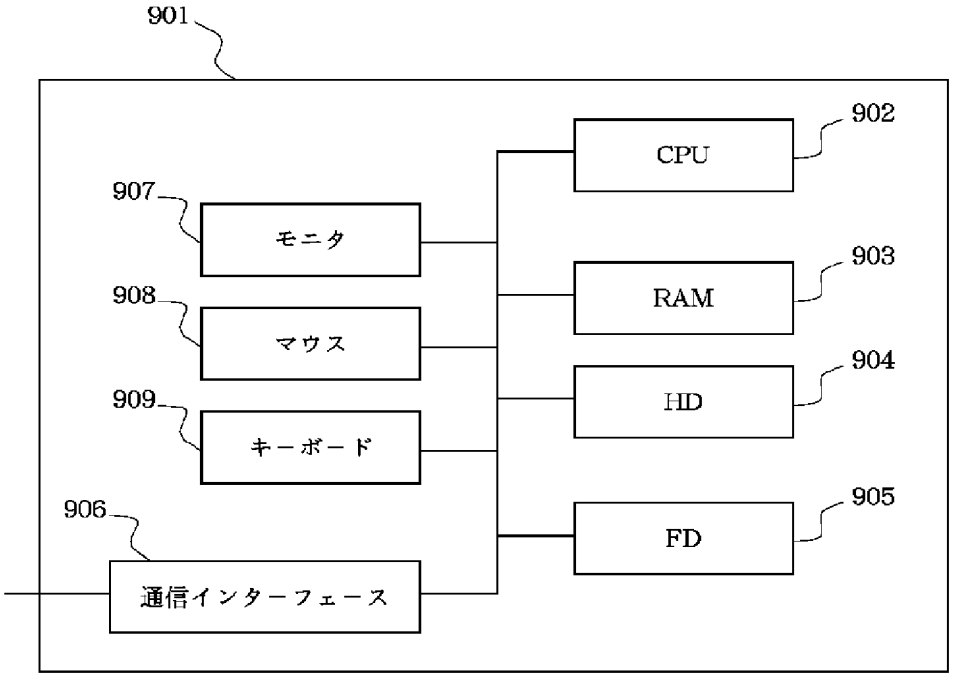
【図１０】



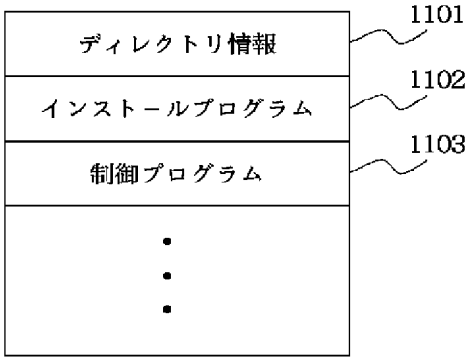
【図３】


(1)
Webアクセススタート
S1
DNS参照
S2
Webサーバ A にアクセス
S3
最適サーバ問い合わせ
S8
アクセスサーバの変更処理
Webサーバ B へアクセス
1
(2)
NS サーバスタート
S4
経路情報問い合わせ
S5
経路情報取得
S6
最適サーバ決定・返答
S7
取得情報蓄積
リターン
(3)
1
S9
Webサーバ B にアクセス
S10
最適サーバ問い合わせ
S13
アクセスサーバの変更処理
Webサーバ C へアクセス
2
(4)
NS サーバスタート
S11
最適サーバ決定
S12
最適サーバ返答
リターン
(5)
2
S14
Webサーバ C にアクセス
S15
最適サーバ問い合わせ
S16
アクセスサーバの変更処理
Webサーバ C へアクセス

【図９】



【図１１】



フロントページの続き

Ｆターム(参考)　5B045 GG02 JJ28
5B089 GA11 GA21 HA01 HA10 HB02
JA11 JA22 JA33 JB02 JB22
KA06 KB04 KB07 MA03 MA07
5K030 GA13 HA06 HB00 HB19 HC01
HD03 JA10 KA02 LB06 MB01
MC07

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2001－290787
（P2001－290787A）
(43)公開日　平成13年10月19日(2001.10.19)

(51)Int.Cl.[7]		識別記号	FI		テーマコード(参考)	
Ｇ０６Ｆ	15/177	６７４	Ｇ０６Ｆ	15/177	６７４Ｂ	５Ｂ０４５
	12/00	５４５		12/00	５４５Ｍ	５Ｂ０８２
	13/00	３５７		13/00	３５７Ｚ	５Ｂ０８９
		５２０			５２０Ｃ	

審査請求　未請求　請求項の数９　ＯＬ　（全　９　頁）

(21)出願番号　特願2000－107892(P2000－107892)

(22)出願日　平成12年４月10日(2000.4.10)

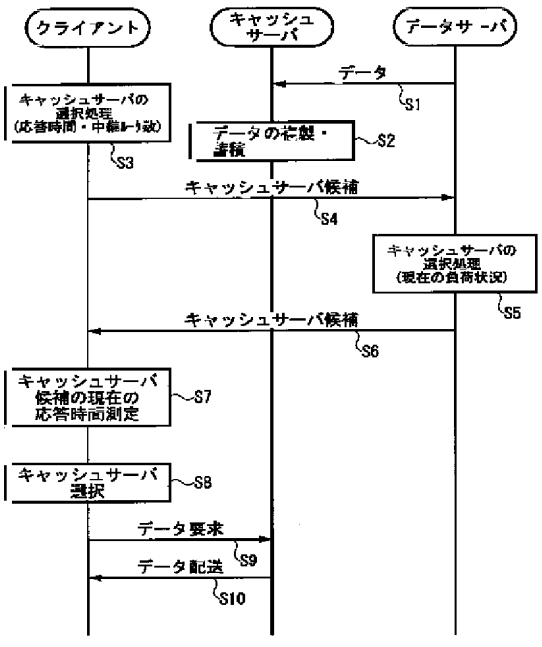
(71)出願人　399035766
エヌ・ティ・ティ・コミュニケーションズ株式会社
東京都千代田区内幸町－丁目１番６号
(72)発明者　荒木　秀教
東京都千代田区内幸町－丁目１番６号　エヌ・ティ・ティ・コミュニケーションズ株式会社内
(74)代理人　100070150
弁理士　伊東　忠彦

最終頁に続く

(54)【発明の名称】　データ配信方法及びデータ配信プログラムを格納した記憶媒体

(57)【要約】
【課題】　ミラーサーバにクライアントからクセスされる頻度が少ない、もしくは、アクセスされることがないデータを蓄積することを避け、ミラーサーバの負荷状況やミラーサーバとクライアント間の回線の混雑状況を考慮して、クライアントへデータを配信するミラーサーバを選択することを可能とするデータ配信方法及びデータ配信プログラムを格納した記憶媒体を提供する。
【解決手段】　本発明は、データサーバからクライアントに直接データを配信せずに、複数のキャッシュサーバの中から、該データ配信時において、負荷が小さく、該クライアントとの通信時間の遅延が小さいキャッシュサーバを選択し、該キャッシュサーバにデータを配信し、キャッシュサーバからクライアントにデータを配信する。

本発明の動作の概要を説明するための図



【特許請求の範囲】
【請求項１】　コンピュータネットワーク経由でデータサーバからクライアントにデータを配信するデータ配信方法において、
前記データサーバから前記クライアントに直接データを配信せずに、複数のキャッシュサーバの中から、該データ配信時において、負荷が小さく、該クライアントとの通信時間の遅延が小さいキャッシュサーバを選択し、該キャッシュサーバにデータを配信し、
前記キャッシュサーバから前記クライアントに前記データを配信することを特徴とするデータ配信方法。
【請求項２】　データを配信するキャッシュサーバを選択する際に、
前記クライアントにおいて、
前記キャッシュサーバの応答時間、及び該クライアントと該キャッシュサーバとの間の中継ルータの数に基づいて、キャッシュサーバを決定する請求項１記載のデータ配信方法。
【請求項３】　データを配信するキャッシュサーバを選択する際に、
前記クライアントでは、予め収集された各キャッシュサーバの応答時間、中継ルータ数の情報に基づいてキャッシュサーバの１次候補を選択して、前記データサーバに送信し、
前記データサーバでは、前記クライアントから取得した前記キャッシュサーバの１次候補から、現在の負荷状況に基づいてキャッシュサーバの２次候補を選択して、該クライアントに送信し、
前記クライアントにおいて、前記データサーバから取得した前記２次候補のキャッシュサーバの現在の応答時間を測定して、該応答時間が最短のキャッシュサーバを、データ配信を行うキャッシュサーバとして決定する請求項１記載のデータ配信方法。
【請求項４】　前記キャッシュサーバから前記クライアントにデータ配信を行う際に、
前記クライアントから要求されたデータが、前記キャッシュサーバに、複製・蓄積されていない場合において、
前記キャッシュサーバの蓄積データ数が一定数を越えている場合には、該キャッシュサーバに蓄積されているデータのうち、最終アクセス時刻、及びアクセス回数に基づいてアクセス頻度の少ないデータを削除し、
前記データサーバからデータを取得して蓄積する請求項１記載のデータ配信方法。
【請求項５】　コンピュータネットワーク経由でクライアントにデータを配信するデータサーバに搭載されるデータ配信プログラムを格納した記憶媒体であって、
前記クライアントに直接データを配信せずに、複数のキャッシュサーバの中から、該データ配信時において、負荷が小さく、該クライアントとの通信時間の遅延が小さいキャッシュサーバを選択し、該キャッシュサーバにデータを配信するプロセスを有することを特徴とするデータ配信プログラムを格納した記憶媒体。
【請求項６】　前記クライアントから取得した、各キャッシュサーバの応答時間、中継ルータ数の情報により選択されたキャッシュサーバの１次候補の現在の負荷状況に基づいて、キャッシュサーバの２次候補を選択して、該クライアントに送信するプロセスを含む請求項５記載のデータ配信プログラムを格納した記憶媒体。
【請求項７】　データサーバから取得したデータをコンピュータネットワーク経由でクライアントにデータを配信するキャッシュサーバに搭載されるデータ配信プログラムを格納した記憶媒体であって、
前記クライアントから要求されたデータが、複製・蓄積されていない場合において、蓄積データ数が一定数を越えている場合には、蓄積されているデータのうち、最終アクセス時刻、及びアクセス回数に基づいてアクセス頻度の少ないデータを削除するプロセスと、
前記データサーバからデータを取得・蓄積するプロセスと、
前記クライアントに要求されたデータを送信するプロセスとを有することを特徴とするデータ配信プログラムを格納した記憶媒体。
【請求項８】　データサーバから取得したデータをコンピュータネットワーク経由で取得するクライアントに搭載されるデータ配信プログラムを格納した記憶媒体であって、
前記キャッシュサーバの応答時間、及び該クライアントと該キャッシュサーバとの間の中継ルータの数に基づいて、キャッシュサーバを決定するプロセスを有することを特徴とするデータ配信プログラムを格納した記憶媒体。
【請求項９】　予め収集された各キャッシュサーバの応答時間、中継ルータ数の情報に基づいてキャッシュサーバの１次候補を選択して、前記データサーバに送信するプロセスと、
前記データサーバから取得した２次候補のキャッシュサーバの現在の応答時間を測定して、該応答時間が最短のキャッシュサーバをデータ配信を行うキャッシュサーバとして決定するプロセスとを有する請求項８記載のデータ配信プログラムを格納した記憶媒体。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、データ配信方法及びデータ配信プログラムを格納した記憶媒体に係り、特に、コンピュータネットワーク経由でサーバからクライアントへデータを配信するためのデータ配信方法及びデータ配信プログラムを格納した記憶媒体に関する。
【０００２】
【従来の技術】比較的容量の大きいデータをコンピュータネットワーク経由でサーバからクライアントへ配信す

る場合、クライアントからの要求に応じてサーバから直接配信すると、複数クライアントからの要求が集中したときサーバの負荷が増大し、データ配信に時間がかかる。

【０００３】また、サーバに負荷が集中していなくても、サーバとクライアントの間の通信回線が混雑していた場合にも、データ配信に時間がかかる。コンピュータネットワークにおいて通信を中継する装置であるルータは、サーバ、クライアント間の通信において経由するルータの数を最小にするように経路を選択する機能が実装されているが、ネットワークの遅延、負荷等のリアルタイムに変化するパラメータに基づく仕組みではない。

【０００４】これらの課題のうちいくつかを解決するため、従来、サーバと同じデータを複製、蓄積し、サーバに変わってデータを配信するミラーサーバというサーバを複数の異なる場所に接続し、クライアントに対して、最寄のミラーサーバからデータ配信を行うようにすることで、サーバへの負荷の集中を回避し、混在した回線を避ける方法が用いられている。

【０００５】

【発明が解決しようとする課題】しかしながら、上記のミラーサーバを用いる方法では、ミラーサーバは、サーバと同じ量のデータを蓄積しなければならない。また、サーバのデータが更新された場合、すべてのミラーサーバにそれを反映しなければならないという問題がある。この場合、サーバの蓄積データ量が増加すれば、ミラーサーバの蓄積データ量も増加し、また、ミラーサーバによってはクライアントからアクセスされる頻度が少ない、もしくは、アクセスされることがないデータも蓄積しておくことになり、サーバからミラーサーバにデータを予め転送しておく処理と転送されたデータを蓄積しておく記憶領域が無駄となる。また、最寄のミラーサーバの選択において、経由するルータの数を基準とする、または、サーバとクライアントの地理的な距離を基準とする等の場合、ミラーサーバの負荷状態や、ミラーサーバとクライアント間の回線の混雑状況は考慮されない。

【０００６】本発明は、上記の点に鑑みなされたもので、ミラーサーバにクライアントからクセスされる頻度が少ない、もしくは、アクセスされることがないデータを蓄積することを避け、ミラーサーバの負荷状況やミラーサーバとクライアント間の回線の混雑状況を考慮して、クライアントへデータを配信するサーバ（キャッシュサーバ）を選択することを可能とするデータ配信方法及びデータ配信プログラムを格納した記憶媒体を提供することを目的とする。

【０００７】

【課題を解決するための手段】本発明（請求項１）は、コンピュータネットワーク経由でデータサーバからクライアントにデータを配信するデータ配信方法において、データサーバからクライアントに直接データを配信せずに、複数のキャッシュサーバの中から、該データ配信時において、負荷が小さく、該クライアントとの通信時間の遅延が小さいキャッシュサーバを選択し、該キャッシュサーバにデータを配信し、キャッシュサーバからクライアントにデータを配信する。

【０００８】本発明（請求項２）は、データを配信するキャッシュサーバを選択する際に、クライアントにおいて、キャッシュサーバの応答時間、及び該クライアントと該キャッシュサーバとの間の中継ルータの数に基づいて、キャッシュサーバを決定する。

【０００９】本発明（請求項３）は、データを配信するキャッシュサーバを選択する際に、クライアントでは、予め収集された各キャッシュサーバの応答時間、中継ルータ数の情報に基づいてキャッシュサーバの１次候補を選択して、データサーバに送信し、データサーバでは、クライアントから取得したキャッシュサーバの１次候補から、現在の負荷状況に基づいてキャッシュサーバの２次候補を選択して、該クライアントに送信し、クライアントにおいて、データサーバから取得した２次候補のキャッシュサーバの現在の応答時間を測定して、該応答時間が最短のキャッシュサーバを、データ配信を行うキャッシュサーバとして決定する。

【００１０】本発明（請求項４）は、キャッシュサーバからクライアントにデータ配信を行う際に、クライアントから要求されたデータが、キャッシュサーバに、複製・蓄積されていない場合において、キャッシュサーバの蓄積データ数が一定数を越えている場合には、該キャッシュサーバに蓄積されているデータのうち、最終アクセス時刻、及びアクセス回数に基づいてアクセス頻度の少ないデータを削除し、データサーバからデータを取得して蓄積する。

【００１１】本発明（請求項５）は、コンピュータネットワーク経由でクライアントにデータを配信するデータサーバに搭載されるデータ配信プログラムを格納した記憶媒体であって、当該データサーバからクライアントに直接データを配信せずに、複数のキャッシュサーバの中から、該データ配信時において、負荷が小さく、該クライアントとの通信時間の遅延が小さいキャッシュサーバを選択し、該キャッシュサーバにデータを配信するプロセスを有する。

【００１２】本発明（請求項６）は、クライアントから取得した、各キャッシュサーバの応答時間、中継ルータ数の情報により選択されたキャッシュサーバの１次候補の現在の負荷状況に基づいて、キャッシュサーバの２次候補を選択して、該クライアントに送信するプロセスを含む。

【００１３】本発明（請求項７）は、データサーバから取得したデータをコンピュータネットワーク経由でクライアントにデータを配信するキャッシュサーバに搭載されるデータ配信プログラムを格納した記憶媒体であっ

て、クライアントから要求されたデータが、複製・蓄積されていない場合において、蓄積データ数が一定数を越えている場合には、蓄積されているデータのうち、最終アクセス時刻、及びアクセス回数に基づいてアクセス頻度の少ないデータを削除するプロセスと、データサーバからデータを取得・蓄積するプロセスと、クライアントに要求されたデータを送信するプロセスとを有する。

【００１４】本発明（請求項８）は、データサーバから取得したデータをコンピュータネットワーク経由で取得するクライアントに搭載されるデータ配信プログラムを格納した記憶媒体であって、キャッシュサーバの応答時間、及び該クライアントと該キャッシュサーバとの間の中継ルータの数に基づいて、キャッシュサーバを決定するプロセスを有する。

【００１５】本発明（請求項９）は、予め収集された各キャッシュサーバの応答時間、中継ルータ数の情報に基づいてキャッシュサーバの１次候補を選択して、データサーバに送信するプロセスと、データサーバから取得した２次候補のキャッシュサーバの現在の応答時間を測定して、該応答時間が最短のキャッシュサーバをデータ配信を行うキャッシュサーバとして決定するプロセスとを有する。

【００１６】上述のように、本発明は、ミラーサーバに蓄積しているデータのうち、クライアントからアクセスされる頻度が少ない、もしくは、アクセスされることがないものを除いたデータを蓄積するサーバ（キャッシュサーバ）をミラーサーバの代わりとしてコンピュータネットワークの異なる位置に複数接続し、サーバの負荷状況やクライアントとの間の回線の混雑状況を考慮して、クライアントへデータを配信するサーバを選択することが可能となる。

【００１７】

【発明の実施の形態】図１は、本発明のデータ配信システムの構成を示す。同図に示すデータ配信システムは、コンピュータネットワーク１０、データサーバ２０、キャッシュサーバ３０、及びクライアント４０から構成される。コンピュータネットワーク１０には、１つのデータサーバ２０と、１つ以上のキャッシュサーバ３０が異なる位置に接続され、クライアント４０は１つ以上接続されるものとする。

【００１８】通常、データサーバ２０は、高速な通信回線によってコンピュータネットワーク１０に接続される。キャッシュサーバ３０は、比較的高速な通信回線によってコンピュータネットワーク１０に接続されるものとする。クライアント４０がコンピュータネットワーク１０に接続される通信回線は、通常の低速回線でもよい。また、キャッシュサーバ３０とクライアント４０は一体化して構築してもよい。

【００１９】データサーバ２０は、データ２１、データ一覧表２２、キャッシュサーバ管理表２３を持つが、キャッシュサーバ３０とクライアント４０が一体化したシステム構成の場合については、後述するキャッシュサーバ選択処理を行わないため、キャッシュサーバ管理表２３は不要となる。データサーバ２０が保持するデータ２１は、データ配信の対象となるすべてのデータの原本である。データ一覧表２２は、データサーバ２０に蓄積されているすべてのデータ名が登録される。

【００２０】キャッシュサーバ管理表２３は、コンピュータネットワーク１０に接続されているすべてのキャッシュサーバ３０について、キャッシュサーバ名、キャッシュサーバアドレス、そのキャッシュサーバが現在データ配信中のクライアント数が登録された表である。キャッシュサーバアドレスとは、例えば、キャッシュサーバのＩＰアドレスを登録する。

【００２１】キャッシュサーバ３０は、データ３１と蓄積データ管理表３２を有する。キャッシュサーバ３０のデータとは、データサーバ２０に蓄積されたデータのうち、一部のデータの複製である。蓄積データ管理表３２は、キャッシュサーバ３０に蓄積された全てのデータについて、データ名、最終アクセス時刻、アクセス回数が登録された表である。データ名とは、そのキャッシュサーバ３０に蓄積されているデータ名を表す。蓄積データ管理表３２の内容は、クライアント４０からのデータ配信要求に応じて更新されるため、キャッシュサーバ３０毎に蓄積管理表３２の内容は異なる。

【００２２】クライアント４０は、キャッシュサーバ一覧表４１を持つが、キャッシュサーバとクライアントが一体化したシステム構成の場合については、後述するキャッシュサーバ選択処理を行わないため、当該キャッシュサーバ一覧表４１は不要となる。キャッシュサーバ一覧表４１は、キャッシュサーバ名、キャッシュサーバアドレス、応答時間、応答時間の更新回数、中継ルータ数から構成される。

【００２３】次に、上記の構成における動作の概要を説明する。

【００２４】図２は、本発明の動作の概要を説明するための図である。

【００２５】キャッシュサーバ３０において、予めデータサーバ２０から一部のデータを取得して（ステップ１）、当該データを蓄積しておく（ステップ２）。なお、キャッシュサーバ３０において、当該処理を行わずに、クライアント４０からの要求があった場合に、データサーバ２０から必要なデータを取得してもよい。また、当該処理は、キャッシュサーバ３０の蓄積データ数が一定の数を超える場合には、最終アクセス時刻やアクセス回数に基づいてアクセス頻度の少ないデータを削除した後に、データサーバ２０からデータを取得するものとする。

【００２６】クライアント４０が、キャッシュサーバ３０を選択する際に、予め収集されている各キャッシュサ

ーバの応答時間、中継ルータ数に基づいて、キャッシュサーバの候補を絞り込み（ステップ３）、当該候補をデータサーバ２０に送信する（ステップ４）。データサーバ２０では、クライアント４０から取得したキャッシュサーバ候補について、現在の当該キャッシュサーバ３０の負荷状況を判断し、負荷が少ないキャッシュサーバ３０をキャッシュサーバ候補とし（ステップ５）、クライアント４０に渡す（ステップ６）。これにより、クライアント４０は、データサーバ２０から渡されたキャッシュサーバ候補について、現在の応答時間を測定し（ステップ７）、応答時間が最短のキャッシュサーバ３０をデータ取得の対象として選択し（ステップ８）、当該キャッシュサーバ３０にデータ配信要求を行う（ステップ９）。これにより、データ配信要求を取得したキャッシュサーバ３０は、当該要求に対応するデータを検索しクライアント４０に提供する（ステップ１０）。但し、要求されたデータが蓄積されていない場合には、前述のように、データサーバ２０から当該データを取得する。

【００２７】

【実施例】以下、図面と共に本発明の実施例を説明する。

【００２８】図３は、本発明のクライアントの準備処理のフローチャートである。これは、図４に後述するデータ配信に先立ってクライアント４０毎に少なくとも１回行うものである。

【００２９】クライアント４０は、データサーバ２０からデータ一覧表の内容を取得する（ステップ１０１）。キャッシュサーバとクライアントが一体化したシステム構成の場合には、後述するキャッシュサーバ選択処理がないため、準備処理は、ここまでとなる。そうでない場合には、クライアント４０は、データサーバ２０からキャッシュサーバ管理表に登録されているキャッシュサーバ名、及びキャッシュサーバアドレスをデータサーバ２０から取得する（ステップ１０２）。クライアント４０において、各キャッシュサーバ３０の応答時間、及び各キャッシュサーバ３０までの中継ルータ数についての情報を収集し、キャッシュサーバ一覧表４１に登録する。応答時間の測定は、例えば、キャッシュサーバ管理表４１のキャッシュサーバアドレスを使用して、クライアント４０から各キャッシュサーバ３０へＩＣＭＰ（インターネット・コントロール・メッセージ・プロトコル（標準化されたプロトコル））のエコーメッセージを送信し、エコー応答メッセージが返るまでの時間をクライアント４０において測定する。各キャッシュサーバ３０までの中継ルータ数は、例えば、キャッシュサーバ管理表のキャッシュサーバアドレスとＲＩＰ（経路情報プロトコル（標準化されたプロトコル））を利用して取得する（ステップ１０３）。キャッシュサーバ一覧表の応答時間の更新回数の初期値は１とする。各キャッシュサーバ３０の応答時間、及び各キャッシュサーバ３０までの中継ルータ数についての情報収集を以前に行ったことがある場合は（ステップ１０４、Ｙｅｓ）、応答時間は以下に示す計算式

（キャッシュサーバ一覧表に登録されている応答時間×キャッシュサーバ一覧表に登録されている応答時間の更新回数＋現在測定したキャッシュサーバの応答時間）÷（キャッシュサーバ一覧表に登録されている応答時間の更新回数＋１）

により、新しい応答時間を算出して更新する。これは、応答時間の平均値である（ステップ１０７）。応答時間の更新回数の値は１増やす（ステップ１０８）。中継ルータ数は以前と異なれば、その値で更新する。

【００３０】また、情報収集を以前に行ったことがない場合は（ステップ１０４，Ｎｏ）、情報収集した応答時間、中継ルータ数をキャッシュサーバ一覧表４１に登録し（ステップ１０５）、キャッシュサーバ一覧表４１の応答時間の更新回数の初期値を１とする（ステップ１０６）。

【００３１】次に、データ配信の流れについて説明する。

【００３２】図４は、本発明のデータ配信処理のフローチャートである。

【００３３】クライアント４０において、データサーバ２０から取得したデータ一覧表を表示し（ステップ２０１）、クライアント４０の利用者がデータを選択する（ステップ２０２）。クライアント４０において、キャッシュサーバ選択処理を行い（ステップ２０３）、決定したキャッシュサーバ３０からクライアント４０へデータ配信処理を行う（ステップ２０４）。キャッシュサーバとクライアントが一体化したシステム構成の場合については、キャッシュサーバ選択処理は行わず、クライアントと一体化したキャッシュサーバからデータ配信処理を行う。

【００３４】次に、上記のステップ２０３におけるキャッシュサーバの選択処理について説明する。

【００３５】図５は、本発明のキャッシュサーバ選択処理のシーケンスチャートである。

【００３６】クライアント４０において、キャッシュサーバ一覧表から応答時間が短い順、同一のものが複数ある場合は、中継ルータ数が少ない順に一定数（例えば、５個）のキャッシュサーバ名を送信する（ステップ３０１）。データサーバ２０において、クライアント４０から取得したキャッシュサーバ名のうち、キャッシュサーバ管理表２３において、現在データ配信中のクライアント数が少ないものについて一定数（例えば、３個）に絞り、そのキャッシュサーバ名をクライアント４０に送信する（ステップ３０２）。クライアント４０において、データサーバ２０から取得したキャッシュサーバ名のキャッシュサーバ３０について、現在の応答時間を測定する（ステップ３０３）。応答時間が最小であるものをデ

ータ配信を行うキャッシュサーバ３０として決定する（ステップ３０４）。クライアント４０において、測定を行ったキャッシュサーバ３０の応答時間は、前述の式を新しい応答時間を算出するために用いて、キャッシュサーバ一覧表４１の応答時間を算出された値に更新する。これは、該当するキャッシュサーバについての現在までの測定値の平均値である。キャッシュサーバ一覧表４１の応答時間の更新回数を１増やす。これらは次回のキャッシュサーバ選択処理において利用される。クライアント４０は、１つに決定したキャッシュサーバ３０に対してデータ配信を要求する（ステップ３０５）。

【００３７】次に、クライアント４０からデータ配信要求を受け付けたキャッシュサーバ３０のデータ配信処理を示す。

【００３８】図６は、本発明の一実施例のキャッシュサーバのデータ配信処理のシーケンスチャートである。

【００３９】クライアント４０は、キャッシュサーバ３０に対してデータ名を指定してデータ配信の要求を行う（ステップ４０１）。そのキャッシュサーバ３０に要求されたデータがすでにデータサーバ２０から複製、蓄積されていれば（ステップ４０２，Ｙｅｓ）、データサーバ２０のキャッシュサーバ管理表２３の該当するキャッシュサーバ名について、現在データ配信中のクライアント数を１増やし（ステップ４０８）、要求されたデータに関する蓄積データ管理表２３の最終アクセス時間を現在の時刻に更新し、アクセス回数を１増やし（ステップ４０９）、キャッシュサーバ３０からデータをクライアント４０へ配信する（ステップ４１０）。配信終了時、データサーバ２０のキャッシュサーバ管理表２３の該当するキャッシュサーバ名について現在データ配信中のクライアント数を１減らす（ステップ４１１）。

【００４０】なお、キャッシュサーバとクライアントが一体化したシステム構成の場合については、キャッシュサーバ選択処理がないため、データサーバ２０は、キャッシュサーバ管理表２３を持たないので、現在配信中のクライアント数増減の操作は行わない。

【００４１】キャッシュサーバ３０に要求されたデータがデータサーバから複製・蓄積されていない場合（ステップ４０２，Ｎｏ）、データサーバ２０からキャッシュサーバ３０へデータを複製、蓄積し（ステップ４０６）、蓄積したデータのデータ名、及びアクセス回数０を蓄積データ管理表３２に登録し（ステップ４０７）、その後、データサーバ２０のキャッシュサーバ管理表２３の現在データ配信中のクライアント数を１増やす（ステップ４０８）。さらに、キャッシュサーバ３０の蓄積データ管理表３２の最終アクセス時間、アクセス回数の更新、クライアント４０へデータ配信、データサーバ２０のキャッシュサーバ管理表２３の現在データ配信中のクライアント数を１減らすことを行う（ステップ４１１）。

【００４２】データサーバ２０からキャスサーバ３０へデータ配信時に、キャッシュサーバ３０の蓄積データ数が一定数を超える場合は、当該キャッシュサーバ３０は、蓄積データ管理表３２により、最終アクセス時刻が古い順、同一のものが複数存在する場合には（ステップ４０３，Ｙｅｓ）、蓄積データ管理表３２により、最終アクセス時刻が古い順、同一のものが複数存在する場合は、アクセス回数が少ない順に削除し、削除したデータに関する登録情報を蓄積データ管理表３２から削除し（ステップ４０４）、その後、データサーバ２０からデータの取得を行う（ステップ４０５）。

【００４３】この方法により、クライアント４０から一度もアクセスされていないキャッシュサーバ３０には、データサーバ２０からデータを複製・蓄積し、クライアント４０からアクセス頻度の高い、一定の数のデータに順次更新され、蓄積されるようになる。

【００４４】本発明は、具体的には、インターネットに接続したパーソナルコンピュータへの音楽データの配信においては、音楽データを蓄積したサーバがデータサーバに相当するが、キャッシュサーバを設け、上記のような仕組みを適用することができる。

【００４５】また、街頭に設置されたキオスク端末と呼ばれる装置へ、利用者の記憶媒体（ＭＤ（ミニディスク）、フラッシュメモリ等）を挿入して、音楽、映像、ゲームソフト等のデータを配信するシステムにおいては、キオスク端末について、上記にあるキャッシュサーバとクライアントが一体化した仕組みを適用することができる。

【００４６】また、上記の各装置（クライアント、データサーバ、キャッシュサーバ）の動作をそれぞれのプログラムとして構築し、それぞれの装置に接続または、内蔵されているメモリや、ディスク装置、または、フロッピー（登録商標）ディスク、ＣＤ－ＲＯＭ等の可搬記憶媒体に格納しておき、本発明を実施する際にインストールすることにより、容易に本発明を実現できる。

【００４７】

【発明の効果】上述のように、本発明によれば、データサーバと同一のデータに対して、複数のクライアントからのアクセスが集中しても、実際にクライアントにデータ配信を行う処理が、複数のキャッシュサーバへ分散される。

【００４８】また、キャッシュサーバは、アクセス頻度の高いデータを蓄積するため、データサーバと同一のすべてのデータを蓄積しなくてもよいため、資源の有効活用が図れる。

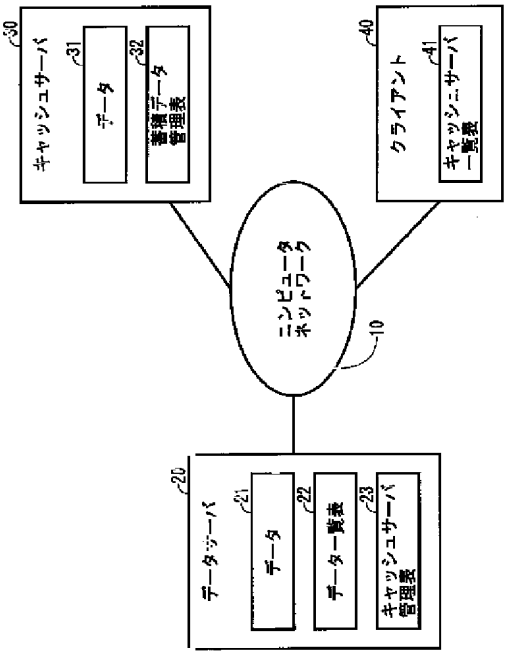
【００４９】また、クライアントでは、キャッシュサーバからのデータ配信時点において、他のキャッシュサーバと比較して負荷が小さく、回線の混雑による通信時間の遅延の小さいキャッシュサーバからデータ配信を受けることができる。

【図面の簡単な説明】
【図１】本発明のデータ配信システムの構成図である。
【図２】本発明の動作の概要を説明するための図である。
【図３】本発明の一実施例のクライアントの準備処理のフローチャートである。
【図４】本発明の一実施例のデータ配信処理のフローチャートである。
【図５】本発明の一実施例のキャッシュサーバ選択処理のシーケンスチャートである。
【図６】本発明の一実施例のキャッシュサーバのデータ配信処理のシーケンスチャートである。

【符号の説明】
１０　コンピュータネットワーク
２０　データサーバ
２１　データ
２２　データ一覧表
２３　キャッシュサーバ管理表
３０　キャッシュサーバ
３１　データ
３２　蓄積データ管理表
４０　クライアント
４１　キャッシュサーバ一覧表

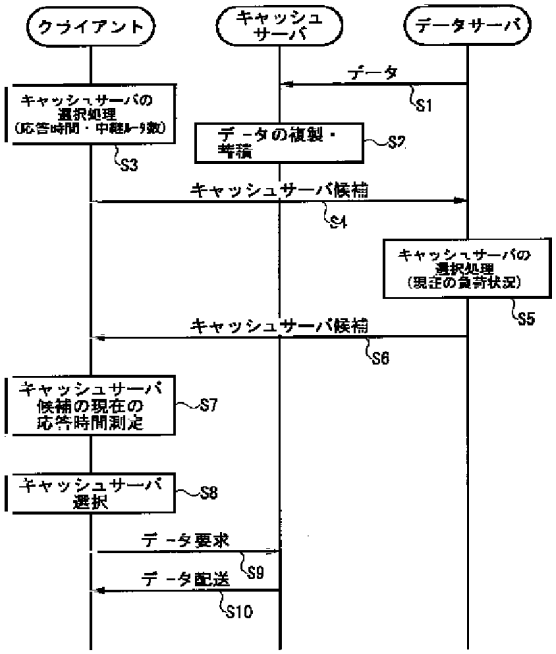
【図１】

本発明のデータ配信システムの構成図



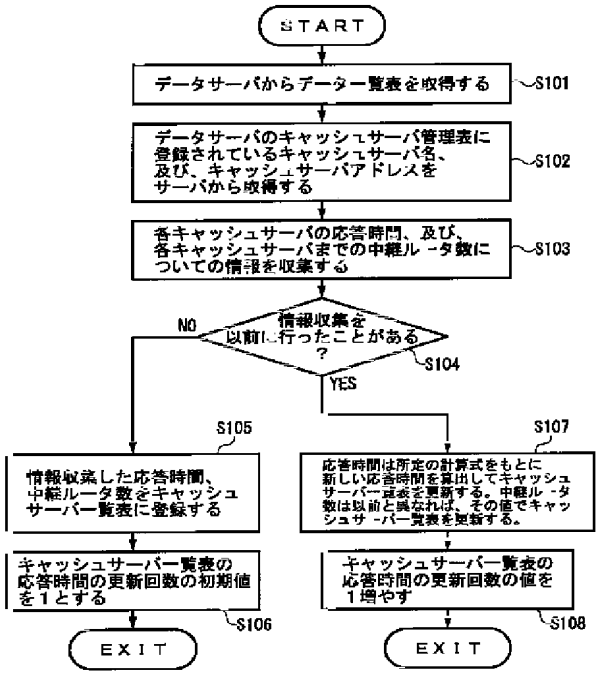
【図２】

本発明の動作の概要を説明するための図



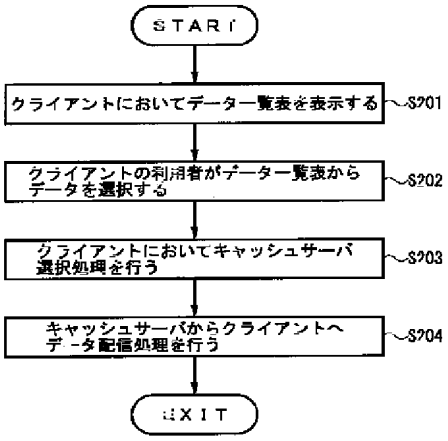
【図3】

本発明の一実施例のクライアントの準備処理のフローチャート



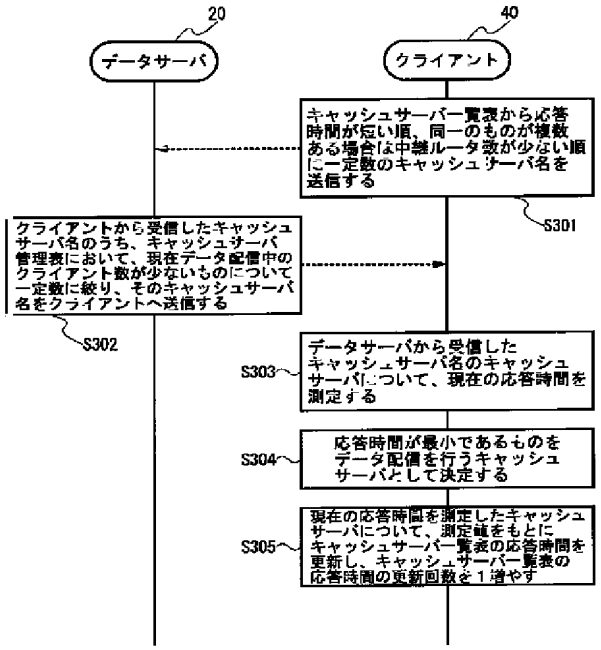
【図4】

本発明の一実施例のデータ配信の処理のフローチャート



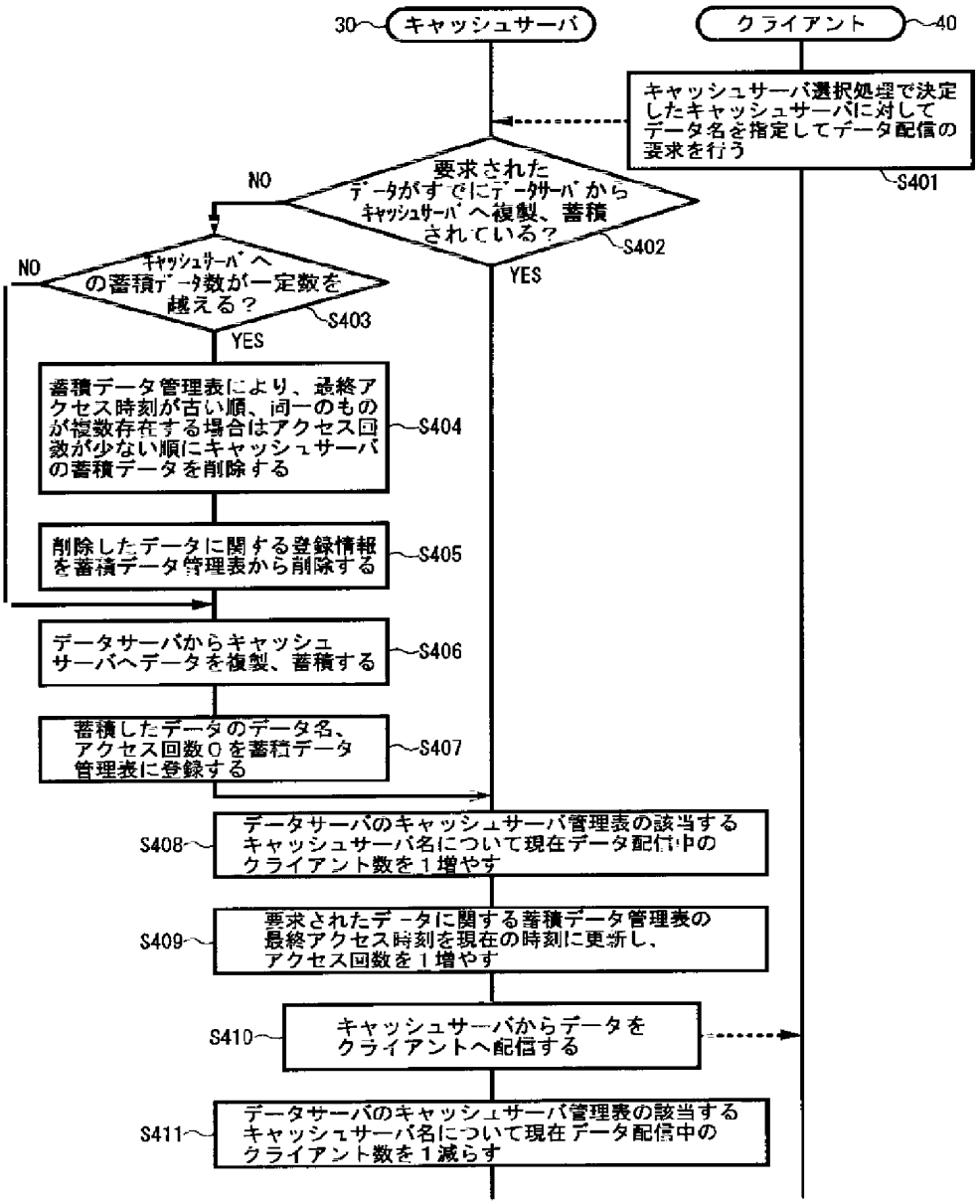
【図5】

本発明の一実施例のキャッシュサーバ選択処理のシーケンスチャート



【図6】

本発明の一実施例のキャッシュサーバのデータ配信処理のシーケンスチャート



フロントページの続き

(72)発明者　野村　進
東京都千代田区内幸町一丁目1番6号　エヌ・ティ・ティ・コミュニケーションズ株式会社内

Fターム(参考)　5B045 GG02 GG04
5B082 FA12 HA05 HA08
5B089 GA11 JA33 JB14 KA06 KA07 KB03 MA03 MA07

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号
特開2001－312434
（P2001－312434A）
(43)公開日　平成13年11月９日(2001.11.9)

(51)Int.Cl.[7]		識別記号	FI			テーマコード(参考)
Ｇ０６Ｆ	13/00	３５７	Ｃ０６Ｆ	13/00	３５７Ｚ	５Ｂ０７５
		５２０			５２０Ｃ	５Ｂ０８２
	12/00	５４５		12/00	５４５Ａ	５Ｂ０８９
	17/30	１１０		17/30	１１０Ｃ	

審査請求　未請求　請求項の数９　ＯＬ　（全　９　頁）

(21)出願番号　特願2000－130203(P2000－130203)

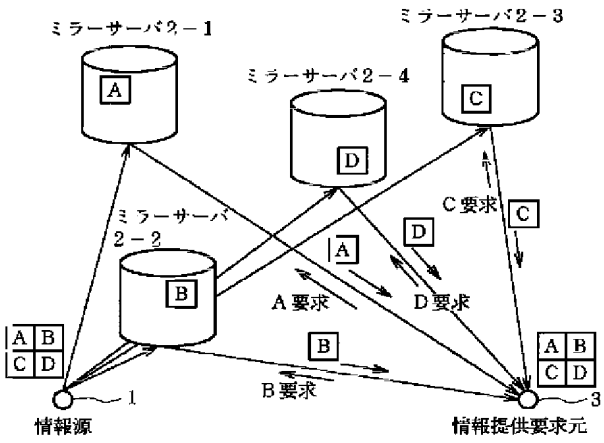
(22)出願日　平成12年４月28日(2000.4.28)

(71)出願人　000004226
日本電信電話株式会社
東京都千代田区大手町二丁目３番１号
(72)発明者　奥川　徹
東京都千代田区大手町二丁目３番１号　日本電信電話株式会社内
(74)代理人　100078237
弁理士　井出　直孝　（外１名）

Ｆターム(参考)　5B075 KK04 PQ05
5B082 DE04 HA08
5B089 GA11 HA10 JA33 KA07 KA12
KC53 ME01

(54)【発明の名称】　情報配信システム

(57)【要約】
【課題】　複数のサーバを効率よく使用することにより高い情報転送能力および耐障害性を実現し、所望の情報を短時間に確実に得ることができる情報配信システムを実現する。
【解決手段】　情報源からの情報を情報ネットワーク上の多地点に設置されたミラーサーバに分割して蓄積しておき、情報提供要求元は、多数のミラーサーバから情報を分割して集める。あらかじめミラーサーバへの情報の蓄積規則を定めておいて自動的に適切なミラーサーバを知る。または、複数の情報の所在を管理する装置を設置し、ミラーサーバと通信することによってそれらが保持する情報を管理し、情報提供要求元がその装置から情報を得て適切なミラーサーバを知る。または、情報源が自らミラーサーバに情報がどのように置かれているかを管理または把握する。さらに、各ミラーサーバの情報提供能力を調べ、それに応じて、分割された情報を集める。



【特許請求の範囲】
【請求項１】　情報を提供する情報源と、情報ネットワーク上の多地点に設置されこの情報源から提供される情報を蓄積する複数のミラーサーバとを備えた情報配信システムにおいて、
前記情報源は、前記情報を複数に分割する手段と、この分割された情報を前記複数のミラーサーバに分散して提供する手段とを備えたことを特徴とする情報配信システム。
【請求項２】　前記分散された情報は、その一部に重複部分を含む請求項１記載の情報配信システム。
【請求項３】　情報を提供する情報源と、情報ネットワーク上の多地点に設置されこの情報源から提供される情報を蓄積する複数のミラーサーバとを備えた情報配信システムにおいて、
前記情報源は、前記複数のミラーサーバにそれぞれ同一の情報を提供する手段を備え、
前記ミラーサーバは、前記情報源から提供された情報のうち情報提供要求された部分を抽出して提供する手段を備えたことを特徴とする情報配信システム。
【請求項４】　前記情報は、欠落部分を相互に補間可能な情報である請求項１または３記載の情報配信システム。
【請求項５】　前記複数のミラーサーバに蓄積された情報内容を管理する手段が設けられ、
この情報内容を管理する手段は、情報提供要求元からの問い合わせに対して所望の情報を保持するミラーサーバを通知する手段を備えた請求項１または３記載の情報配信システム。
【請求項６】　前記情報内容を管理する手段は、前記情報源と併せて設置された請求項５記載の情報配信システム。
【請求項７】　前記情報内容を管理する手段は、情報提供要求にしたがって所望の情報を保持するミラーサーバから情報を取得してこの情報提供要求元に提供する手段を備えた請求項５記載の情報配信システム。
【請求項８】　前記複数のミラーサーバのそれぞれについて情報提供能力を管理する手段が設けられ、
この情報提供能力を管理する手段の提供能力情報にしたがって所望の情報提供を要求するミラーサーバを選択する手段を備えた請求項１または３あるいは７記載の情報配信システム。
【請求項９】　前記選択する手段は、前記提供能力情報にしたがって選択したミラーサーバに対する情報提供要求順序を設定する手段を含む請求項８記載の情報配信システム。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は情報ネットワークにおける情報配信に利用する。本発明は、情報ネットワークにおける大量あるいは高い信頼性を要する情報配信に利用するに適する。
【０００２】
【従来の技術】情報ネットワークにおいて大量の情報を多数の受信者に配信する必要がある場合には、情報を蓄積して提供する装置（以下サーバと呼ぶ）に、極めて高い処理能力を持つハードウェアを用意するか、あるいは複数のサーバを組み合わせる方式が用いられる。また、サーバの能力が十分でない場合や、情報伝達経路が輻輳する可能性が高い場合、あるいは十分な情報転送帯域を確保できない場合には、ミラーサーバと呼ばれる同一の情報を有する複数のサーバを異なる情報経路上に用意する方式が用いられる。
【０００３】
【発明が解決しようとする課題】サーバの処理能力を高くする方法では電気回路的な限界が存在し、ある一定以上の能力を有するためには大きなコストが必要となる。また、それを複数のサーバを同一地点に複数設置する方式で回避したとしても、そのサーバにアクセスするための回線に負荷が集中することや、何らかの故障が生じてそのサーバ群からの情報伝達が不可能になった場合には、情報を得ることができなくなるといった問題がある。
【０００４】一方、ミラーサーバを設置する方式では、同一の情報を有するサーバを多くの地点に設置する必要があるが、情報提供要求元は通常そのうちの１つにしかアクセスしないため、各サーバにおける負荷を偏らせ十分な情報転送能力を発揮できない可能性が存在する。
【０００５】本発明は、このような背景に行われたものであって、複数のサーバを効率よく使用することにより高い情報転送能力および耐障害性の情報配信システムを提供することを目的とする。本発明は、所望の情報を短時間に確実に得ることができる情報配信システムを提供することを目的とする。
【０００６】
【課題を解決するための手段】本発明では、情報源からの情報を情報ネットワーク上の多地点に設置されたミラーサーバに分割して蓄積しておき、情報提供要求元は、多数のミラーサーバから情報を分割して集めることにより所望の情報を得る。このように、直接情報を得るミラーサーバを複数にすることにより、一ヶ所に大きな能力を有するサーバを設置する必要がない。また、複数のミラーサーバから情報を得ることにより、一部のミラーサーバに負荷が集中するような事態を避けることができる。このとき、情報を分割して配信すれば各ミラーサーバにおける情報の蓄積量を節約できる。
【０００７】一方、ミラーサーバに重複して情報を置けば、同一情報に対するミラーサーバへのアクセスを分散できると共に、特定のミラーサーバに故障が発生するなどして情報の提供能力が低下あるいは失われる場合に、

他のミラーサーバが持つ情報で補うことができる。これは、インターネットにおけるコンテンツへのアクセスのように、複数の情報提供要求元が情報を得ようとする場合に、情報に対する大量の提供要求が同時に発生してサーバにおける情報提供能力が低下するなどの場合にも有効である。

【０００８】このような場合に、情報提供要求元は、一定の時間制限を超えて情報が得られなかったミラーサーバへの要求をあきらめ、他のサーバへと改めて要求を行う。あるいは、あらかじめ取得する情報に冗長性を持たせ、不足した情報を冗長な情報から導出することも可能である。

【０００９】ところで、情報が複数のミラーサーバに置かれている場合には、情報提供要求元が情報の所在地を知ることは重要である。本発明では、あらかじめミラーサーバへの情報の蓄積規則を定めておいて自動的に適切なミラーサーバを知ることができるようにしたり、複数の情報の所在を管理する装置を設置し、ミラーサーバと通信することによってそれらが保持する情報を管理し、情報提供要求元がその装置から情報を得て適切なミラーサーバを知ることができるようにしたり、または、情報源が、自らが提供する情報がミラーサーバにどのように置かれているかを管理または把握し、情報提供要求元からの要求に対してどのミラーサーバにアクセスすべきかを返答するようにすることもできる。さらに、代行して情報を収集し、情報提供要求元に提供する装置を設置し、それが情報の所在を管理するようにすることもできる。

【００１０】また、各ミラーサーバはそれぞれ情報提供能力、また各時点における負荷が一般には異なる。したがって各ミラーサーバに同程度の情報提供を要求していては高い性能が得られない可能性がある。そこで、各ミラーサーバの情報提供能力を調べ、それに応じて、分割された情報を集めることで効率よく所望の情報を得られる。情報提供能力は、信号の往復時間から判断したり、あるいは、ある程度の時間までに得られた情報量からフィードバックして判断することができる。このようにして判断した情報提供能力に応じて最終的に要求する情報量を加減することができる。

【００１１】すなわち、本発明は、情報を提供する情報源と、情報ネットワーク上の多地点に設置されこの情報源から提供される情報を蓄積する複数のミラーサーバとを備えた情報配信システムである。

【００１２】ここで、本発明の特徴とするところは、前記情報源は、前記情報を複数に分割する手段と、この分割された情報を前記複数のミラーサーバに分散して提供する手段とを備えたところにある。これにより、一ヶ所に大きな能力を有するサーバを設置する必要がない。また、複数のミラーサーバから情報を得ることにより、一部のミラーサーバに負荷が集中するような事態を避けることができる。このとき、情報を分割して配信すれば各ミラーサーバにおける情報の蓄積量を節約できる。

【００１３】このように、情報が複数のミラーサーバに置かれている場合には、情報提供要求元が情報の所在地を知ることは重要である。これについては、例えば、あらかじめミラーサーバへの情報の蓄積規則を定めておけば自動的に適切なミラーサーバを知ることができる。

【００１４】前記分散された情報は、その一部に重複部分を含むこともできる。あるいは、前記情報源は、前記複数のミラーサーバにそれぞれ同一の情報を提供する手段を備え、前記ミラーサーバは、前記情報源から提供された情報のうち情報提供要求された部分を抽出して提供する手段を備えることもできる。

【００１５】これにより、同一情報に対するミラーサーバへのアクセスを分散できると共に、特定のミラーサーバに故障が発生するなどして情報の提供能力が低下あるいは失われる場合に、他のミラーサーバが持つ情報で補うことができる。このような場合に、情報提供要求元は、一定の時間制限を超えて情報が得られなかったミラーサーバへの要求をあきらめ、他のサーバに改めて要求を行うことができる。

【００１６】前記情報は、欠落部分を相互に補間可能な情報であることが望ましい。このように、あらかじめ取得する情報に冗長性を持たせておけば、不足した情報を冗長な情報から導出することもできる。

【００１７】前記複数のミラーサーバに蓄積された情報内容を管理する手段が設けられ、この情報内容を管理する手段は、情報提供要求元からの問い合わせに対して所望の情報を保持するミラーサーバを通知する手段を備える構成とすることもできる。これにより、情報提供要求元が適切なミラーサーバを知ることができる。

【００１８】前記情報内容を管理する手段は、前記情報源と併せて設置されることもできる。これによれば、情報源が、自らが提供する情報がミラーサーバにどのように置かれているかを管理または把握し、情報提供要求元からの要求に対してどのミラーサーバにアクセスすべきかを返答することができる。

【００１９】前記情報内容を管理する手段は、情報提供要求にしたがって所望の情報を保持するミラーサーバから情報を取得してこの情報提供要求元に提供する手段を備えた構成とすることもできる。このように、代行して情報を収集し、情報提供要求元に提供する手段を設置し、それが情報の所在を管理するようにすることもできる。

【００２０】前記複数のミラーサーバのそれぞれについて、情報提供能力を管理する手段が設けられ、この情報提供能力を管理する手段の提供能力情報にしたがって所望の情報提供を要求するミラーサーバを選択する手段を備えた構成とすることもできる。これにより、効率よく所望の情報を得ることができる。情報提供能力は、信号

の往復時間から判断したり、あるいは、ある程度の時間までに得られた情報量からフィードバックして判断することができる。このようにして判断した情報提供能力に応じて最終的に要求する情報量を加減することができる。

【００２１】また、前記選択する手段は、前記提供能力情報にしたがって選択したミラーサーバに対する情報提供要求順序を設定する手段を含むことが望ましい。これにより、選択したミラーサーバの中でもさらに提供能力のレベルにしたがって情報提供要求を行う順序を設定することができる。例えば、提供能力が比較的高いミラーサーバについては所望の一連の情報の先頭から多数の情報提供を要求し、提供能力が比較的低いミラーサーバについては所望の一連の情報の末尾から少数の情報提供を要求する。このようにすれば、この二つのミラーサーバから一連の情報をその先頭および末尾から同時に取得することができるため、情報取得を効率良く速やかに行うことができる。

【００２２】

【発明の実施の形態】本発明実施例の情報配信システムの構成を図１、図２、図３、図４、図５、図６、図９、図１０、図１１、図１２、図１３を参照して説明する。図１は本発明第一実施例の情報配信システムの全体構成図である。図２および図３は本発明第二実施例の情報配信システムの全体構成図である。図４は本発明第二実施例における一部のミラーサーバに障害が発生した場合の動作を説明するための図である。図５は本発明第三実施例における一部のミラーサーバに障害が発生した場合の動作を説明するための図である。図６は本発明第四実施例の情報配信システムの全体構成図である。図９は本発明第六実施例の情報配信システムの全体構成図である。図１０は本発明第七実施例の情報配信システムの全体構成図である。図１１は本発明第八実施例の情報配信システムの全体構成図である。図１２および図１３は本発明第九実施例における情報提供要求の動作を説明するための図である。

【００２３】本発明は、情報を提供する情報源１と、情報ネットワーク上の多地点に設置されこの情報源１から提供される情報を蓄積する複数のミラーサーバ２－１～２－４とを備えた情報配信システムである。

【００２４】ここで、本発明の特徴とするところは、図１に示すように、情報源１は、前記情報を複数に分割し、この分割された情報を複数のミラーサーバ２－１～２－４に分散して提供するところにある。

【００２５】図２に示すように、前記分散された情報は、その一部に重複部分を含むこともできる。あるいは、図３に示すように、情報源１は、複数のミラーサーバ２－１～２－４にそれぞれ同一の情報を提供し、ミラーサーバ２－１～２－４は、情報源１から提供された情報のうち情報提供要求された部分を抽出して提供することもできる。これにより、図４に示すように、一部のミラーサーバに障害が発生したときに、正常な他のミラーサーバにより対処することができる。

【００２６】また、前記情報は、欠落部分を相互に補間可能な誤り訂正符号を用いた情報とすることもできる。これにより、図５に示すように、一部のミラーサーバに障害が発生したときに正常なミラーサーバから提供された情報により、本来、障害を起こしたミラーサーバから提供されるべき情報を導出することができる。

【００２７】また、図９に示すように、複数のミラーサーバ２－１～２－４に蓄積された情報内容を管理する手段である情報管理装置６が設けられ、この情報管理装置６は、情報提供要求元３からの問い合わせに対して所望の情報を保持するミラーサーバを通知する。

【００２８】図１０に示すように、情報管理装置６を情報源１と併せて一つの装置として構成し、情報提供管理装置７－１および７－２とすることもできる。

【００２９】また、図１１に示すように、情報管理装置６は、情報提供要求にしたがって所望の情報を保持するミラーサーバから情報を取得してこの情報提供要求元３に提供する情報取得管理装置８とすることもできる。

【００３０】図１２および図１３に示すように、情報提供要求元３は、複数のミラーサーバ２－１～２－４のそれぞれについて、情報提供能力を管理し、提供能力情報にしたがって所望の情報提供を要求するミラーサーバを選択することもできる。また、選択したミラーサーバに対する情報提供要求の順序を設定することができる。以下では、本発明実施例をさらに詳細に説明する。

【００３１】（第一実施例）本発明第一実施例を図１を参照して説明する。本発明第一実施例の情報配信システムは、情報源１および情報ネットワーク上の多地点に設置された情報を蓄積して提供する複数のミラーサーバ２－１～２－４および情報提供要求元３からなる。情報源１からの情報を情報ネットワーク上の多地点に設置されたミラーサーバ２－１～２－４に分割して蓄積しておき、情報提供要求元３が、ミラーサーバ２－１～２－４から分割された情報を集めることで所望の情報を得る。

【００３２】図１の例では、情報をＡ、Ｂ、Ｃ、Ｄの４つに分割し、それぞれをミラーサーバ２－１、２－２、２－３、２－４に蓄積しておき、情報提供要求元３がそれらのミラーサーバ２－１～２－４から情報を取得することにより元の情報を復元して得ることができる。

【００３３】情報の分割方法としては、パケット単位、ブロック単位、ファイル単位の分割等が考えられる。ビットあるいはバイト単位の分割も方式的には可能であるが、ネットワークの異なる地点にあるミラーサーバからの情報の到達時刻には一般的にばらつきがあるため、分割単位が細かすぎて効率が悪いものと考えられる。

【００３４】パケット単位の分割においては、情報をあらかじめパケットネットワークに転送する形にしてお

き、そのパケット単位で分割して保存しておき、情報提供要求元３が複数のミラーサーバ２－１～２－４からパケット単位で情報を取得し、元の情報に復元する方式などがある。

【００３５】通常の１対１のＴＣＰ／ＩＰ通信方式においては、パケット到着順序が守られる必要があるが、この方式においては到着順序が入れ替わる可能性があるため、情報提供要求元３において通信手順を一部変更し、バッファを設けるなどの対策をする必要がある。

【００３６】パケットより大きいブロック単位の分割を用いる場合においては、情報提供要求元３側でまずブロック単位の情報の復元を行い、それから情報全体を復元する。アプリケーション側あるいはミドルウェアとしてブロック単位のファイルを復元する機構が必要である。ファイル単位の分割においては、複数のファイルが１つのまとまりのある情報として構成されている場合に、その構成要素となるファイル単位で異なるミラーサーバにアクセスする。この場合には、同一のファイルは１つのサーバから取得するため、既存のアプリケーションに対する変更点が少ない。これは、例えばＷＷＷのように、一度に画面表示するオブジェクトとして複数のファイルが存在する場合などに適用可能である。

【００３７】（第二実施例）本発明第二実施例を図２ないし図４を参照して説明する。図２および図３は、図１と同様の構成であるが、図２では、各ミラーサーバ２－１～２－４が有する情報はそれぞれ他のミラーサーバの所有する情報と一部重複するようになっている。図３では、各ミラーサーバ２－１～２－４はそれぞれ情報全体を有する。

【００３８】このようにしておくことにより、図４に示すようにいずれかのミラーサーバに障害が発生したり性能が著しく低下して情報を提供できなくなった場合に、他のミラーサーバを利用することで欠けた情報を補うことが可能である。図４で示す例では、ミラーサーバ２－２に障害が発生し、分割された情報の１つである情報Ｂの提供が不可能になっている。このとき、情報提供要求元３では情報Ｂをミラーサーバ２－１に改めて要求することによりそれを取得する。

【００３９】（第三実施例）本発明第三実施例を図５を参照して説明する。第二実施例と同様に、一部のミラーサーバに障害が発生した場合の対策としては、図５に示すような例も挙げられる。これは、分割された情報に冗長性を持たせ、ある程度の情報欠落に対しては得られた情報の演算により導出できるようにするものである。図５の例では、分割された情報Ａ、Ｃ、Ｄには冗長性があり、Ｂが欠けても残りの情報Ａ、Ｃ、Ｄの関数としてＢに相当する情報を導出している。このような導出方式は、誤り訂正符号を用いる方式などがあり、例えば、誤り訂正符号の一つであるリードソロモン符号を用いパケット単位での情報欠落を補完可能であることなどが示されている。

【００４０】（第四実施例）本発明第四実施例を図６を参照して説明する。複数のミラーサーバ２－１～２－４を設置する場合には、ミラーサーバ２－１～２－４へ情報を送ること自体が負荷となる可能性もある。これに対しては同時に複数のミラーサーバ２－１～２－４に情報を伝達可能な方式を用いることで解決できる。すなわち、インターネットプロトコル（ＩＰ）レイヤにおいてはＩＰマルチキャスト方式を使用するものである。

【００４１】一方で、より低位の物理レイヤで情報を分配する方式を用いることも可能である。これらにおいて、ミラーサーバが一部の情報だけを保持し、全ての情報を保持しない場合には、各ミラーサーバにおいて不要な情報は破棄される。図６はそれを模式的に示した例である。情報源１は情報を１回しか送出しないが、途中にある情報分配装置４により全てのミラーサーバ２－１～２－４へ情報を送出できる。

【００４２】情報分配装置４として、ＩＰマルチキャスト方式としてはレイヤ３の転送機能を有するルータ等が相当し、光信号を用いる物理層においては光カプラを用いることができる。

【００４３】（第五実施例）本発明第五実施例を図７および図８を参照して説明する。図７は本発明第五実施例の情報源からミラーサーバへの情報転送の様子を示す図である。図８は本発明第五実施例の情報提供要求元３が情報を取得する様子を示す図である。図７に示す例では、リング構成のネットワークを分配装置として用いたものである。情報源１はリング構成のネットワークに対してネットワーク接続装置５－１を介して情報を送出し、リング構成のネットワークでは、情報はリングを周回しながら各ミラーサーバ２－１～２－４に情報が転送される。情報提供要求元３が情報の要求を送ると、各ミラーサーバ２－１～２－４は自分に割り当てられた情報をネットワーク接続装置５－２を介して送出し、情報提供要求元３に届ける。

【００４４】（第六実施例）本発明第六実施例を図９を参照して説明する。これまで述べた例では、情報提供要求元３は適切なミラーサーバに向けて情報の要求を行わなければならない。そのための方式として、例えばキャッシュの方法として知られるハッシュを用いてファイル名等の属性とその情報を保有するミラーサーバをあらかじめ規則付けておく方法も適用可能である。

【００４５】本発明第六実施例では、図９に示すように情報の所在を管理する情報管理装置６を設置する。情報管理装置６は、常にミラーサーバ２－１～２－４と通信を行い、所有する情報を認識している。情報提供要求元３は、得ようとする情報に対して、どのミラーサーバへアクセスすべきか情報管理装置６に照会し、その返答を得て適切なミラーサーバへ要求を出す。

【００４６】情報管理装置６が情報を管理する方法とし

ては、定期的にミラーサーバ２－１～２－４に対して問い合わせを行う方法、ミラーサーバ２－１～２－４からその所有する情報の一覧もしくは差分の情報を送信してもらう方法、また、情報源１からミラーサーバ２－１～２－４への情報の蓄積時に、同時に情報管理装置６へ通知する方法などが可能である。

【００４７】（第七実施例）本発明第七実施例を図１０を参照して説明する。図１０に示す例では、情報源１が自らが提供する情報についてはミラーサーバ２－１～２－４にどのように保存されているかについても管理または把握する情報提供管理装置７－１および７－２が設置される。情報提供要求元３は、まず情報提供管理装置７－１に問い合わせを行い、情報提供管理装置７－１がミラーサーバ２－１～２－４に関する情報を提供することにより、最寄りあるいは最も高速に転送が可能等の条件にしたがってミラーサーバの組み合わせを選択して情報を取得する。

【００４８】別の情報提供管理装置７－２の情報を読み出す場合はその情報提供管理装置７－２に問い合わせを行う。この例では、各情報提供管理装置７－１および７－２は自らが提供する情報についてだけ管理すれば良く、情報提供要求元３はまず各情報提供管理装置７－１および７－２に問い合わせを行えばその情報の所在地についての情報が得られるため、管理・問い合わせが容易である。

【００４９】（第八実施例）本発明第八実施例を図１１を参照して説明する。図１１に示す例では、情報の所在を管理し情報を代行して取得する情報取得管理装置８が設置される。情報提供要求元３は、情報の所在を管理し情報を代行して取得する情報取得管理装置８に情報を要求する。情報の所在を管理し情報を代行して取得する情報取得管理装置８は自らが常に情報の所在を管理し、情報提供要求元３からの要求に対して適切なミラーサーバに必要な量の情報を要求し、元の情報を復元した後情報を情報提供要求元３へ転送する。

【００５０】この例では、情報提供要求元３は複数の装置に情報の要求を出す必要がないため、既存のアプリケーションに対して整合性が良い。なお、情報の所在を管理し情報を代行して取得する情報取得管理装置８の代わりに、情報を代行して取得する装置と情報の所在を管理する装置を個別に用意してもよい。

【００５１】（第九実施例）本発明第九実施例を図１２および図１３を参照して説明する。各ミラーサーバ２－１～２－４はそれぞれ情報提供能力また各時点における負荷が一般には異なる。したがって、常に各ミラーサーバ２－１～２－４に均等な量の情報提供を要求していては、高い性能が得られないこともありうる。そこで、効率よく所望の情報を得るため、各ミラーサーバ２－１～２－４の情報提供能力を調べ、それに応じて、分割された情報を集める方式が可能である。

【００５２】各ミラーサーバ２－１～２－４の情報提供能力は、信号の往復にかかる時間や、ある程度の時間までに得られた情報量などから判断あるいはあらかじめデータベースに登録しておく方法などが考えられる。信号の往復時間から判断する場合には、何らかの信号に対して返答を調べ、その時間が少ないものに対して多い量の情報を要求するが、これは概ね物理的に近い距離で負荷が少ないミラーサーバを選択することになる。

【００５３】図１２はそれを模式的に示した例であり、ミラーサーバ２－１の方がミラーサーバ２－２よりも信号の往復時間が短いものとする。このとき、情報提供要求元３は、その度合いから判断して、早い返答が得られたミラーサーバ２－１から情報Ａ、Ｂ、Ｃを得て、返答が遅いミラーサーバ２－２からは情報Ｄを得るようにしている。また、ある一定の時間までに得られた情報量で判断する方法では、最初少ない量の情報を要求し、それに要する時間から、単位時間あたりの情報送出量を計算し、最も少ない時間で全ての情報が得られるようにそれ以降の情報の要求先を調整する。

【００５４】必要に応じ逐次その計算を繰り返し、要求先を調整することによって、最終的に最短の時間で全ての情報を得る。あるいは、複数のサーバに一部重複する要求を送り、全ての情報を得た段階で要求を停止し、重複して得た情報は破棄する方法も可能である。図１３はそれを模式的に示した例であり、Ａ～Ｆに分割された情報について、ミラーサーバ２－１からは分割した情報Ａ、Ｂ、Ｃ、…と取得しつつ、ミラーサーバ２－２から情報Ｆ、Ｅ、…と取得し、全部が揃ったところで終了する。終了の通知のタイミングによっては、重複するデータが到着する場合があり、この例ではミラーサーバ２－１から情報Ｅを取得することもあり得るが、それは破棄すればよい。

【００５５】また、第八実施例で説明した情報取得管理装置８に、第九実施例で説明した各ミラーサーバ２－１～２－４の情報提供能力を調べ、それに応じて、分割された情報を集める方式を適用することもできる。

【００５６】

【発明の効果】以上説明したように、本発明によれば、複数のサーバを効率よく使用することにより高い情報転送能力および耐障害性の情報配信システムを実現することができる。これにより、所望の情報を短時間に確実に得ることができる。

【図面の簡単な説明】

【図１】本発明第一実施例の情報配信システムの全体構成図。

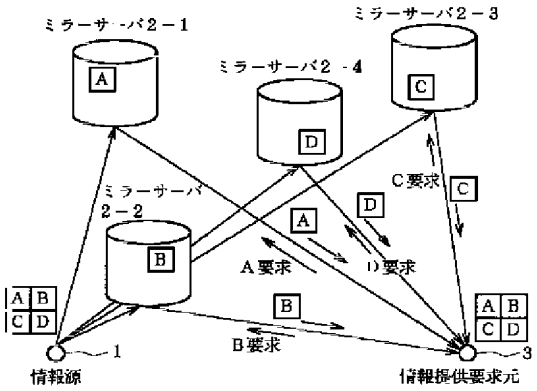
【図２】本発明第二実施例の情報配信システムの全体構成図。

【図３】本発明第二実施例の情報配信システムの全体構成図。

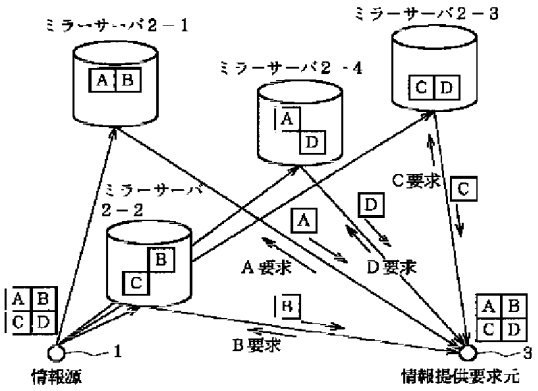
【図４】本発明第二実施例における一部のミラーサーバ

に障害が発生した場合の動作を説明するための図。
【図５】本発明第三実施例における一部のミラーサーバに障害が発生した場合の動作を説明するための図。
【図６】本発明第四実施例の情報配信システムの全体構成図。
【図７】本発明第五実施例の情報源からミラーサーバへの情報転送の様子を示す図。
【図８】本発明第五実施例の情報提供要求元が情報を取得する様子を示す図。
【図９】本発明第六実施例の情報配信システムの全体構成図。
【図１０】本発明第七実施例の情報配信システムの全体構成図。
【図１１】本発明第八実施例の情報配信システムの全体構成図。
【図１２】本発明第九実施例における情報提供要求の動作を説明するための図。
【図１３】本発明第九実施例における情報提供要求の動作を説明するための図。
【符号の説明】
１　情報源
２－１～２－４　ミラーサーバ
３　情報提供要求元
４　情報分配装置
５－１、５－２　ネットワーク接続装置
６　情報管理装置
７－１、７－２　情報提供管理装置
８　情報取得管理装置

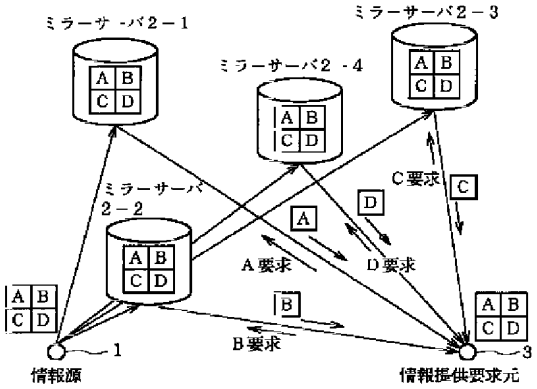
【図１】



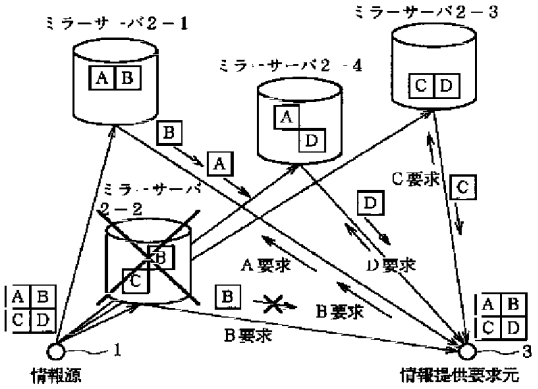
【図２】



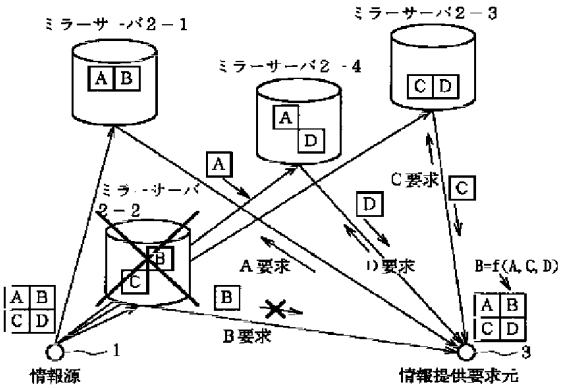
【図３】



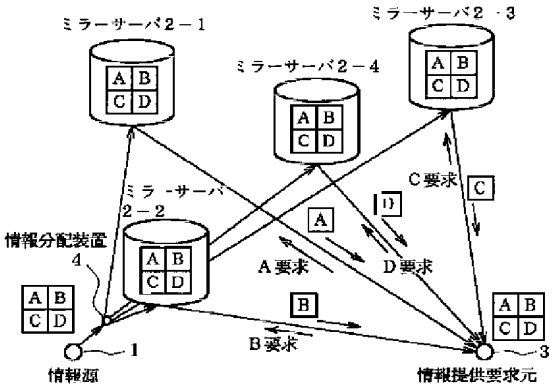
【図４】



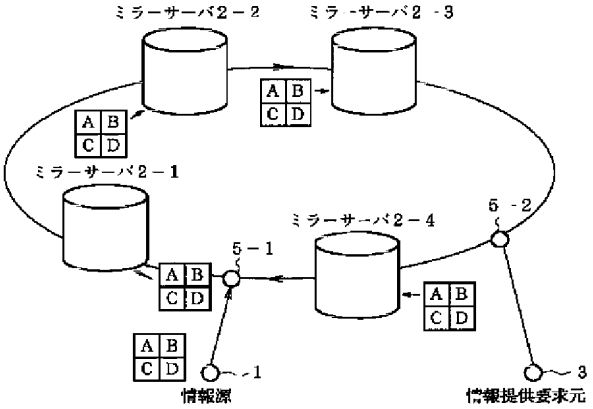
【図5】



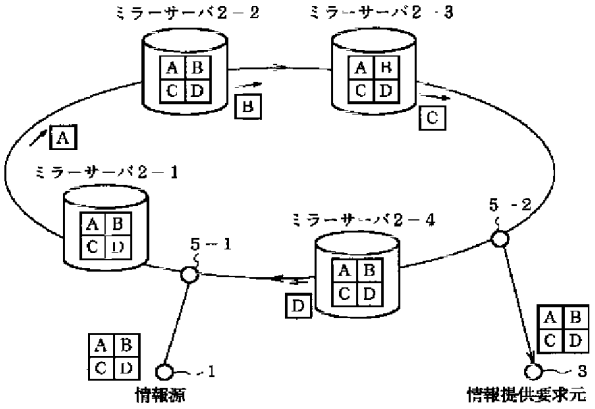
【図6】



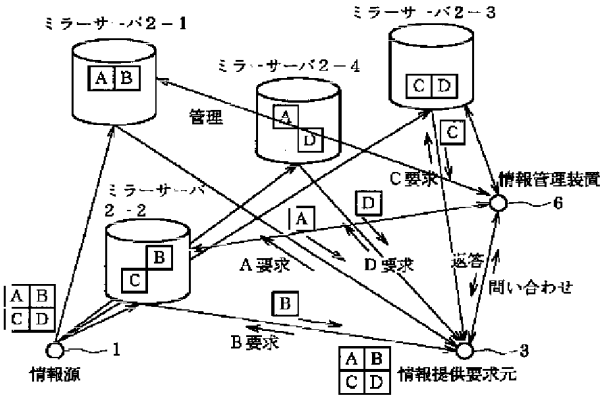
【図7】



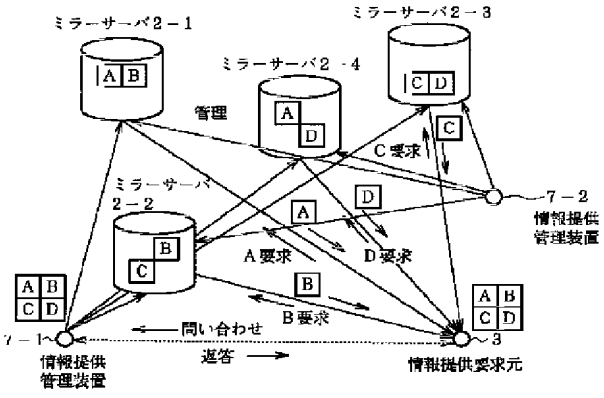
【図8】



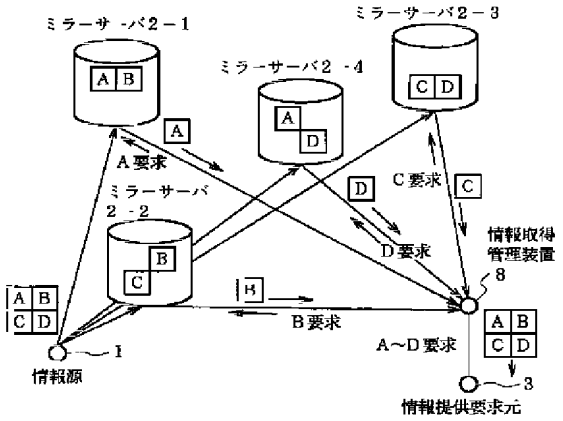
【図9】



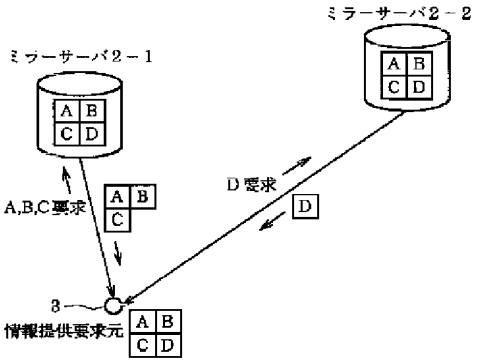
【図10】



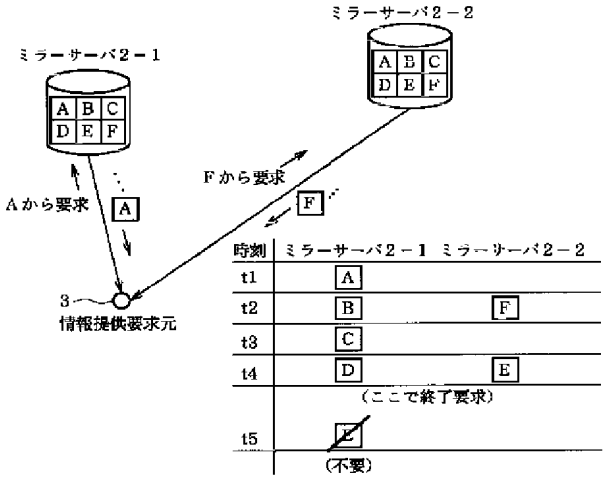
【図１１】


ミラーサーバ2－1
ミラーサーバ2－3
ミラーサーバ2－4
ミラーサーバ2－2
A要求
C要求
D要求
B要求
A～D要求
情報取得管理装置
8
情報源
1
情報提供要求元
3

【図１２】


ミラーサーバ2－1
ミラーサーバ2－2
A,B,C要求
D要求
情報提供要求元
3

【図１３】


ミラーサーバ2－2
ミラーサーバ2－1
Fから要求
Aから要求
情報提供要求元
3
時刻
ミラーサーバ2－1
ミラーサーバ2－2
t1
t2
t3
t4
(ここで終了要求)
t5
(不要)

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
6 December 2001 (06.12.2001)

PCT

(10) International Publication Number
WO 01/93533 A2

(51) International Patent Classification[7]: H04L 29/06

(21) International Application Number: PCT/US01/17799

(22) International Filing Date: 1 June 2001 (01.06.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

60/209,007	1 June 2000 (01.06.2000)	US
09/665,204	18 September 2000 (18.09.2000)	US
09/867,868	29 May 2001 (29.05.2001)	US

(71) Applicant: **AEROCAST.COM, INC.** [US/US]; 6450 Sequence Drive, San Diego, CA 92121 (US).

(72) Inventors: **RACIBORSKI, Nathan, F.**; 470 Arapaho Drive, Jackson, WY 83002 (US). **THOMPSON, Mark, R.**; 7240 N. Dreamy Draw Drive, #104, Phoenix, AZ 85020 (US).

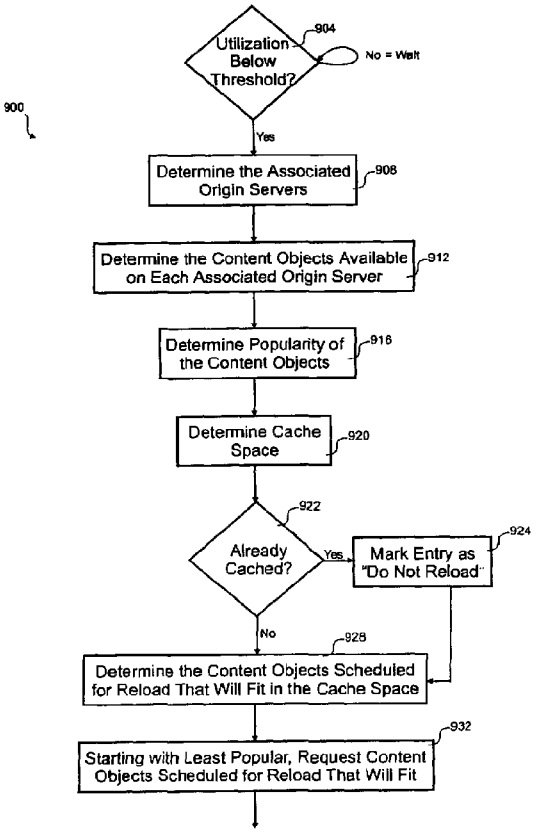
(74) Agents: **FRANKLIN, Thomas, D.** et al.; Townsend and Townsend and Crew LLP, Two Embarcadero Center, Eight Floor, San Francisco, CA 94111-3834 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian

[Continued on next page]

(54) Title: PRELOADING CONTENT OBJECTS ON CONTENT EXCHANGES



(57) Abstract: According to the invention, a way for loading content objects in a content store on a network is disclosed. In one step, processing waits for a triggering event. It is determined if a first origin server is authorized to store content in the content store. A first content object is loaded from the first origin server onto the content store without a request for the first content object. It is determined if a second origin server is authorized to store content in the content store. A second content object is loaded from the second origin server onto the content store without a request for the second content object.

WO 01/93533 A2



patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

Published:

— *without international search report and to be republished upon receipt of that report*

PRELOADING CONTENT OBJECTS ON CONTENT EXCHANGES

This application claims the benefit of U.S. Provisional Application No. 60/209,007 filed on June 1, 2000 and U.S. Patent Application No. 09/665,204 filed on September 18, 2000.

BACKGROUND OF THE INVENTION

This invention relates in general to storing information and, more specifically, to storing information on a network.

There is an ongoing effort in the broadband industry to improve quality of service (QOS) for streaming content. Today, the Internet infrastructure is ill-equipped to provide streams of data at a constant data rate over long periods of time. One solution some have tried is to cache information at some point between the user requesting the data and the origin server providing the data.

There are many other limitations to the Internet infrastructure that affect QOS. An example of a current limitation that affects QOS are the limitations imposed by Internet service providers (ISPs) on upstream bandwidth. Typically, upstream bandwidth is limited to a fraction of the downstream bandwidth by the Internet service provider.

Limited upstream bandwidth is further constrained by peak loading conditions in oversubscribed bandwidth situations. For example, there is an increased demand on Internet systems just after work that is generally attributed to large amounts of people checking e-mail and browsing the web. This increased demand taxes all users such that each of their network connections is slow. Anything that can reduce bandwidth required during those peak periods is greatly desired.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is described in conjunction with the appended figures:

FIG. 1 is a block diagram of an embodiment of a content distribution system;

FIG. 2 is a block diagram of an embodiment of an active directory portion of the content distribution system;

FIG. 3A is a block diagram of an embodiment of an origin server portion of the content distribution system;

FIG. 3B is a block diagram of an embodiment of an external origin server portion of the content distribution system;

FIG. 4A is a block diagram of an embodiment of a content exchange portion of the content distribution system;

FIG. 4B is a block diagram of another embodiment of a content exchange portion of the content distribution system;

FIG. 4C is a block diagram of an embodiment of a content exchange site including multiple content exchange servers;

FIG. 5 is a block diagram of an embodiment of a client computer portion of the content distribution system;

FIG. 6 is a block diagram of an embodiment of a content distribution system;

FIG. 7 is a block diagram of another embodiment of the content distribution system that has two types of content exchanges;

FIG. 8 is a block diagram of yet another embodiment of the content distribution system;

FIG. 9 is a flow diagram of an embodiment of a process for populating the captive content exchange that has available storage capacity;

FIG. 10 is a flow diagram of another embodiment of a process for populating the captive content exchange;

FIG. 11 is a flow diagram of yet another embodiment of a process for populating the captive content exchange;

FIG. 12 is a flow diagram of an embodiment of a process for populating a distributed storage system with content from an origin server; and

FIG. 13 is a flow diagram of an embodiment of a process for providing a list of content exchanges to a client computer.

In the appended figures, similar components and/or features may have the same reference label. Further, various components of the same type may be distinguished by following the reference label by a dash and a second label that distinguishes among the similar components. If only the first reference label is used in the specification, the description is applicable to any one of the similar components having the same first reference label irrespective of the second reference label.

DESCRIPTION OF THE SPECIFIC EMBODIMENTS

The ensuing description provides preferred exemplary embodiment(s) only, and is not intended to limit the scope, applicability or configuration of the invention. Rather, the ensuing description of the preferred exemplary embodiment(s) will provide those skilled in the art with an enabling description for implementing a preferred exemplary embodiment of the invention. It being understood that various changes may be made in the function and arrangement of elements without departing from the spirit and scope of the invention as set for in the appended claims.

The present invention includes techniques for preloading content exchanges with content objects. Typically, the content exchanges that are preloaded are associated with client computers and origin servers. Some or all of the content objects on origin servers are loaded on the associated content exchanges. A triggering event, such as low bandwidth utilization into the content exchange, causes the content exchange to load content objects.

Referring to FIG. 1, a block diagram of an embodiment of a content distribution system 100 is shown. In this embodiment, the content distribution system 100 includes an active directory 104, one or more origin servers 108, one or more client computers 112, one or more content exchanges 116, one or more external origin servers 118, one or more unassociated origin servers 106, the Internet 120 and a crawling directory 124. A particular client computer 112 interacts with the active directory 104 to select a content object for download. The object can be played during download if it is streaming media or can be stored for playback at a later time. The content object could be any type of information, such as audio, video or data, that is available for download from a network. The request for the content object is forwarded to the appropriate origin server 108 along with preference information from the client computer 112. The origin server 108 decides where the object is downloaded from. In order to provide sufficient QOS, any of the content exchanges 116 or even the origin server 108 itself could provide the object.

The active directory 104 can be the interface that the client computer 112 uses for selecting a content object. Software for both the origin server 108 and optionally for the client computer 112 can be downloaded from the active directory 104 to enable the content distribution system 100. Either a directory interface page or a search interface page in the active directory 104 may be used to determine the content object desired for download. The interfaces are maintained in an active manner to avoid broken links to

content objects on the origin servers 108. When a content object is requested from the origin server 108 by a content exchange 116, the active directory 104 can provide a path back to the proper origin server 108 that originated the content object.

Other embodiments could have multiple active directories. Users of the system could be divided amongst the several active directories to distribute the loading among the active directories. Additionally, the other active directories could be used for redundancy such that if one active directory were offline, the others would absorb the loading.

In some embodiments, the origin server 108 serves as the source of a content object, directs a user to a preferred source of the content object and provides directory information to the active directory 104. Content objects are introduced to the system 100 by origin servers 108. Introduction involves selection by an origin server administrator of the content objects to make available to the active directory 104. The administrator is person or system that manages the origin server 108. The content objects could include previously stored information or a streaming feed of information. According to a predetermined cycle, the origin server 108 provides a catalog of the selected information that is updated as the content on the origin server 108 changes.

The origin server 108 determines the preferred source to direct the client computer 112 to in order to download the content object. The preference list of the client computer 112, the loading of the content exchanges and the location of copies of the content object are all considerations of the origin server 108 in redirecting the client computer to the preferred source of the information. That source could be the origin server 108 itself or one of the content exchanges 116.

The user directs the client computer 112 to the location to find the desired content object and subsequently download that object. Using viewer object proxy software downloaded from the active directory 104, the client computer 112 determines the content exchanges 116 that can deliver content with the adequate QOS. The process of determining a content exchange 116 with adequate QOS involves, for example, receiving test information from the content exchanges 116 who are likely to produce the best results and preparing a preferred list based upon those test results. The user can modify the preferred list of content exchanges 116 if a customized approach is desired. When the origin server 108 is deciding the source of the requested content object, the preference information is used to provide adequate QOS.

Unassociated content servers 106 are servers on the Internet that are not formally associated with the system 100. In this embodiment, the content exchanges 116 will still cache the content objects from the unassociated content servers 106 to provide enhanced QOS to the client computers 112. In some embodiments, the users of the client computers 112 will compensate the content exchange owners for the enhanced QOS.

The external origin servers 118 can be additional sources of content objects available to the client computer 112. In an embodiment, external origin servers 118 are coupled to a content exchange(s) 116. The content objects on the external origin servers are mirrored on the distributed content exchanges to enhance distribution without overly taxing the external origin server. Client computers 112 requesting content objects from an external origin server 118 is transparently redirected to a content exchange 116 to fulfill the request for that content object.

The content exchanges 116 are storage repositories that cache content objects. A number of these content exchanges 116 are distributed to different points of the Internet 120 to cache content objects. In this way, any client computer 112 is likely to have a cache close to it on the Internet 120. Information can be cached based upon a number of considerations, such as the desirability of information to users, as a service to origin servers 108 who want their content readily available to users, as a service to users who want improved QOS, or to offload upstream bandwith. Grouping of the content exchanges 116 could be in clusters or individually to service the demand of client computers 112 for content objects.

As stated above, an origin server 108 may refer a client computer 112 request for a content object to a content exchange 116 for fulfillment of the request. When a requested content object or part of a requested content object is not found by a user requesting it from a content exchange 116, a request by the content exchange 116 to other content exchanges is made for that content object. If no other content exchanges 116 have the content object, the active directory 104 is queried for the origin server 108, unassigned origin server 106, or external origin server 188 who is the source of the content object and the content object is downloaded from there to the content exchange 116. While the content exchange 116 is gathering the content object, the client computer 112 is receiving any initial portions of the content object that are available. The content object could be stored in pieces on several content exchanges 116 and, under those circumstances, the requesting content exchange 116 will retrieve those pieces to reassemble the whole content object as needed by the client computer 112.

A crawling directory 124 is used to supplement the catalog information reported by all the origin servers 108. When searching for content objects for a user, the active directory 104 could display content objects available from the origin servers 108 as well as other content objects uncovered by the crawling directory 124 that resides on unassociated origin servers 106 or external origin servers 118. By traversing the web, crawling directories 124 catalog the content objects and other information they encounter. For example, a keyword search of the crawling directory 124 by way of the active directory 104 can direct the user to a content object on an external origin server 118 or unassociated origin server 106 that is not available from any origin server 108. One of the content exchanges 116 could be selected by the active directory 104 to cache the content object as the client computer 112 downloads it. Although only one crawling directory 124 is used in this embodiment, other embodiments could use a number of crawling directories to offer additional search results.

The Internet 120 is comprised of servers linked together by nodes. Data is divided into packets that travel through the Internet 120 by hopping from one node to the next until the destination is reached. Each packet may take a different route through the Internet and arrive at the destination at a different time. Additionally, some packets can be lost during travel through the Internet 120 as the bandwidth of any node saturates. As the number of hops between the source of a content object and the destination increases, so does the likelihood of excessive delay and packet loss.

As the content object traverses a path from source to destination through the Internet 120, the smallest bandwidth between any two nodes in the path defines the maximum bandwidth of that path. Generally, the bandwidth from the Internet 120 to the client computer (i.e., the "last mile") has the least bandwidth allocation. At other times, however, some other hop between nodes has the smallest bandwidth. Caching the content object in a content exchange 116 with a minimal amount of hops between the content exchange 116 and the client computer 112 improves the likelihood of adequate or sufficient QOS.

Downloading the content object at a desired data-rate that does not exceed the data-rate of the client computer to the Internet 120 is adequate QOS. The maximum QOS a user can expect is defined by the speed of their network connection, the processing power of their computer and other factors. The minimum QOS is subjectively defined by the user based upon the quality they desire within certain limits. For example, a user with a 400 Kbps network connection and a fast computer may have a choice of a 28 Kbps, 56

Kbps, or 128 Kbps stream for an audio clip from which the user chooses the 128 Kbps stream. So long as the datarate provided by the client computer is in the range of 128 - 400 Kbps, adequate QOS is possible for that stream.

Although the above embodiment primarily uses the Internet 120 as a connection between the various blocks, other embodiments could use private links outside or in parallel to the Internet 120. The private links could allocate bandwidth sufficient for improved QOS or use other techniques to improve QOS. Additionally, content objects on external origin servers 118 or unassociated origin servers 106 could benefit from the system 100 to the extent caching, encryption and compression is provided.

With reference to FIG. 2, a block diagram of an embodiment of an active directory portion 104 of the content distribution system 100 is shown. Included in the active directory 104 are a dynamic domain name server (DNS) 204, a server manager 208, a directory page 212, a search page 216, a download page 220, a subscriber database 224, a server database 228, a content exchange database 232, and a route database 234. The content distribution system 100 interacts with the active directory 104 to provide directory information to the user and assist in downloading a content object to the user.

Both the user of the client computer 112 and the administrator of the origin server 108 are subscribers to the content distribution system 100. Software is downloaded from a download page 220 to the user and/or administrator. The software for the client computer 112 is optional in some embodiments and improves QOS when installed. The software for the origin server 108 allows the active directory 104 to update the content available on the system 100 and to direct the client computer 112 to a preferred source for receiving that content.

The client computer 112 attaches to the domain of the active directory 104 to find a desired content object. Depending on preference, the user may use a directory page 212 or search page 216 to find the content object. The search page 216 may be a traditional boolean search engine that accesses a catalog of the content objects provided by all origin servers 108 as well as information gathered from the crawling directory 124. Other embodiments could only display information from the crawling directory 124 after a search of the content from the origin servers 108 is unsuccessful or omit information from the crawling directory 124 altogether. The catalog of content objects for all origin servers 108 is maintained in the server database 228.

The directory page 212, in this embodiment, organizes the possible content objects in a hierarchy of categories that are organized by subject. For example, the first

page might show a number of topics of general interest where the user selects sports. In the next page, which is one level down in the hierarchy, a number of sports are displayed where the user selects football. Down another level in the hierarchy, the user may select the San Diego Chargers™ to see another page of related content object links. Each link points to the origin server 108 that originated the content object.

The administrator categorizes the content on the origin server 108 to allow the directory page 212 to present it properly. On a site, directory or file basis, the administrator can choose a category for content objects in an HTML SSI tag associated with that content object. This classification is harvested and stored on the active directory to allow presenting content objects in different categories. Additionally, a moderator may describe and arrange content objects in the categories for the directory page 212. For example, the moderator could mark certain content objects for more prominent display and/or add a review and/or synopsis for the content objects.

The server manager 208 maintains information on all client computers 112, all origin servers 108, all external origin servers 118, all content exchanges 116, and all content objects on origin servers 108. The information related to client computers 112 and origin servers 108 is maintained in the subscriber database 224. The full name, a login name, a password, a unique identifier, token credits available, and other information is maintained in the subscriber database 224 for each user associated with a client computer 112. This database 224 also holds the last time the origin server 108 was verified, an Internet Protocol (IP) address for the origin server 108, the port the content manager server runs upon, on-/off-line status of the origin server 108, a banner ad URL, a name for the origin server 108, a description of the origin server 108, the credits or tokens needed to use the origin server 108 or other billing model, and the number of connections or viewers allowed at one time.

Information on content objects for all origin servers 108 is maintained in the server database 228. For each content object, the origin server name, content object file name and path are stored along with category information, a brief description and keywords. The server database 228 is queried to provide content selections to the user during navigation of the directory and search pages 212, 216. To maintain current information in the server database 228, the server manager 208 periodically interacts with the origin server 108 to get the most recent changes to the catalog of content objects and to determine if the origin server 108 has gone offline. Whenever an origin server 108

goes offline, the entries in the server database 228 corresponding to that origin server 108 are removed and the status information in the subscriber database 224 is updated.

In some embodiments, the entries in the server database 228 remain even after the origin server 108 goes offline. The status is updated to reflect that the content associated with the origin server is unavailable, but the information remains stored in the server database 228. If the status is updated to online, the information is once again presented to users that are searching for content objects. In some circumstances, an origin server 108 may indicate to the active directory 104 that it is going offline for a period of time. Presuming the period of time is short, the active directory can keep information in the server database 228 without presenting it to users.

A list of the content exchanges 116 available to the system 100 is maintained by the server manager 208 in the content exchange database 232. In some embodiments, the content exchange database 232 can include a list of IP addresses for all possible content exchanges 116 within the content distribution system 100. Further, the content exchange database 232 can include a number of content exchange fields associated with each content exchange 116. For example, fields associated with each content exchange 116 in the content exchange database 232 can include a content exchange identifier, a content exchange site, a content exchange provider, a content exchange name, a content exchange location, a content exchange status, an icon, or any other desired information.

Together, the content exchange identifier and content exchange site identify a unique content exchange 116 at a particular content exchange site. The content exchange provider is an indicator of the party responsible for the content exchange, such as XYZ company. The content name is a domain name and the content exchange location can be geographic coordinates for a content exchange 116. In an exemplary embodiment, the content exchange database 232 includes a content exchange identifier, a content exchange site, a content exchange IP address, a content exchange provider, a content exchange name, a content exchange location, and an icon for each content exchange 116 in the content exchange database 232.

In general, the content exchange database 232 can be regionalized or partitioned by a variety of methods including, but not limited to, improved QOS criteria or subscription services criteria. Additionally, the content exchange database 232 can include a listing of alternative active directories 104, origin servers 108, or any other useful information.

The content exchanges 116 in the system 100 regularly provide status to the server manager 208. As content exchanges 116 become available or unavailable, their operational status is reported to the server manager 208 and recorded in the content exchange database 232. In some embodiments, the content exchange database 232 can include additional status information including content exchange 116 loading, capacity, utilization, health, etc.

The routing database 234 includes a list of external origin servers 118. In some embodiments, the routing database 234 includes identification, status and organization information related to the external origin servers 118. Identification information can include an IP address and a domain name for an external origin server 118. Status information can include availability, loading or other status about the external origin server. Organization information can include a list of alternative external origin servers 118. In an embodiment, the routing database 234 includes an IP address and a domain name for each external origin server 118. In some embodiments, identifying an external origin server 118 in the routing database 234 is the mechanism for associating the external origin server 118 with the content distribution system 100.

For each external origin server 118 in the routing database, there are a list of user names that are allowed to use a content exchange 116 to access the information on the external origin server 118. The user name is unique to the user of the client computer 112. A list of external origin servers 118 a client computer can route through a content exchange 116 is provided to the client computer 112. The client computer 112 uses the information from the routing database to redirect user requests for an external origin server 118 to a content exchange 116. After the content exchange 116 populates with the content objects from the external origin server 118, bandwidth is offloaded from the external origin server 118 to the content exchange 116. The user may compensate the owner of the content exchange 116 and/or the administrator of the origin server 108 for this enhanced service.

The dynamic DNS 204 provides an origin server name for each IP address for the origin servers 108. The origin server name uniquely identifies the origin server 108 on the Internet 120. This information is maintained in the subscriber database 224. The content exchange 116 does not know the IP address of the origin server 108 that provided the content object to the content exchange 116, but knows the origin server name. When a content exchange 116 wants to populate its cache with a content object or a portion of a content object that is not available from other content exchanges 116, the

dynamic DNS 204 is queried to determine the IP address or domain name of the origin server 108 that is the source of the content object. If a domain name is retrieved from the dynamic DNS 204, the IP address corresponding to that domain name is retrieved from a DNS.

Referring next to FIG. 3A, a block diagram of an embodiment of an origin server portion 108 of the content distribution system 100 is shown. The origin server 108 is managed by an administrator and provides one source of content objects to the content distribution system 100. QOS is provided by the origin server 108 directing the client computer to a content exchange 116 that can adequately deliver the desired content object. Included in the origin server 108 are a content source 304, a content server 308, a content manager 312, a local content catalog 316, a content location database 320, content exchange information 324, and a health check 330.

Content is provided to the origin server 108 by a content source 304. The content source 304 could include a live web cam, a video or audio feed, a data object, a data stream, a video tape or audio tape, an optical or magnetic disk, or any other content delivery mechanism. Content objects are delivered by the content source 304 to the content server 308 for possible distribution to the system 100.

Time and date information is maintained in each content exchange 116 for the content objects or portions of content objects maintained therein. The time and date information allows distinguishing content objects which may have the same origin sever name, path name and file name. Other embodiments could alternatively use any unique code such as a checksum, CRC or hash to uniquely identify a content object.

All content objects of the origin server 308 are stored on the content server 308. The administrator can select a content object or groups of content objects for publishing to the system 100 while leaving other content objects on the content server 308 such that they are unavailable to the system 100. Some content objects are discreet files, but others are streams of content produced, for example, by live web cams. In some embodiments, the software that runs the content server 308 may be integrated with the software of the content manager 312.

The content manager 312 publishes the desired content objects to the system 100 and directs users to the preferred content exchange 116 for downloading content objects associated with the content manager 312. At the direction of the administrator, the content manager 312 selects content objects or groups of content objects by filename, directory or drive volume for publishing to the active directory 104.

Some content objects on the content server 308 may be excluded from publishing such that they are not available to the system 100.

The content objects selected for publishing to the system are maintained in a local content catalog 316. Entries in the local content catalog 316 are kept current by the content manager 312 as the objects corresponding to those entries may become unavailable or updated. For each entry, the content object file name and path are stored along with category information, a brief description and keywords. Upon attachment to the system 100, the local content catalog 316 is sent by the content manager 312 to the active directory 104 for entry to the server database 228. Periodically, changes to the local content catalog 316 are sent to the server database 228 to keep the directory information as current as possible. Updates could happen on a regular interval such as every two minutes and/or whenever a change in the local content catalog is made.

The content manager 312 also knows the location of all portions of content objects associated with that content manager 312. Upon attaching to the system 100, the content manager 312 contacts each of the content exchanges 116 for status. Each of the content exchanges 116 periodically reports on its content object or content object portions to the originating content managers 312. Armed with this information, the content manager 312 can direct a client computer 112 to a content exchange 116 that may have some or all of the desired content object available for download.

Location information for content objects that is reported by all the content exchanges 108 is maintained by the content manager in the content location database 320. By querying the content location database 320, the content manager 312 can determine the content exchanges 116 that contain a content object or a portion of a content object. During the routing of the client computer 112 to a content source, the presence of the content object in a particular content exchange 112 can affect the routing determination.

The content exchange information store 324 holds information on all active content exchanges 116. Upon power-up of the origin server 108, the content exchange database 232 in the active directory 104 is downloaded into the content exchange information store 324. All the content exchanges 116 listed in the content exchange information store 324 are queried after power-up for status information that is retained as content exchange information 324. The status information includes the number of concurrent links used, total number of concurrent links allowed, bandwidth utilization, and cache churn rate. The churn rate of the cache is the amount of time unused data remains in the cache and is indicative of the loading of the cache. For

example, data is usually flushed out of the cache quickly for content exchanges 116 that are busy relative to the amount of storage in the cache. However, some embodiments that could have sticky content objects that remains pinned in the content exchange 116 for a period of time regardless of use.

The content manager 312 intelligently redirects the client computer 112 wanting a content object to the preferred source for that object. Preference information provided from the client computer 112 is used to determine the sources of the content object preferred by the client computer 112. This information is used along with the current locations of the content object and the loading of the possible content exchanges in order to direct the client computer 112 to the preferred content exchange 116 for download of the content object.

In some embodiments, the content manager 312 can regulate access to content objects. When a client computer 112 attempts to download a content object associated with a content manager 312, a login dialog can be presented if the administrator has secured the content object. The user may enter a user name and/or password in the login dialog to enable redirection of the client computer 112 to a source for the content object. This user name and/or password is in addition to any required for the active directory 104. Before redirecting the client computer 112 to that source, the user name and/or password or login information is checked against a list of acceptable login information previously stored on the origin server 108. Access to a whole origin server 108 or a volume, a directory or a content object on the origin server 108 may be regulated in this manner.

Some embodiments allow the origin server 108 or active directory 104 to preload content objects on a content exchange 116. The requests for content objects are monitored to determine desirability. Desirability information, billing information, size, and/or other considerations are used to determine which content objects to preload on a content exchange 116. Either the origin server 108, active directory 104 or content exchange 116 can request the content objects from a content exchange 116 to preload them there. Periodically, the content objects could be requested for a content exchange 116 to keep them loaded on the content exchange 116 such that they are not unloaded because of inactivity. Other embodiments could designate the content objects as sticky content objects as described above.

The health check 330 can be either a hardware or software application which provides operational characteristics of an associated origin server 108. In an

embodiment, the health check 330 provides a single indication of origin server 108 status. The single indication is a normalized value between zero and one indicating a combination of origin server characteristics. For example, it could include a CPU load, a CPU temperature, a number of concurrent connections, and/or a number of requests an origin server is facilitating. In an alternate embodiment, the health check 330 could monitor characteristics of a content exchange 116 while running on another content exchange 116 or origin server 108.

Referring next to FIG. 3B, a block diagram of an embodiment of an external origin server portion 118 of the content distribution system 100 is shown. The external origin server 118 differs from the origin server 108 in that the external origin server 118 does not have content manager software installed upon it. Included in the external origin server 118 are a content server 308 and a content source 304.

The administrator of the external origin server 118 determines one or more client computers 112 authorized for receiving content objects through a content exchange 116. The routing database 234 is updated by the administrator to allow individual client computers 112 to access the content objects through a content exchange 116. A web page on the active directory provides an interface to entering information on client computers 112 into the routing database 234. Other embodiments could automate the interface between the external origin server 118 and the routing database 234 such that users are automatically added to the routing database 234 upon signing up with the external origin server 118.

The updates to the routing database 234 are downloaded and stored locally by client computer 112. Subsequent attempts to access the external origin server 118 are redirected to a content exchange 116 to service that request. Redirection in this way allows an external origin server 118 to redirect client computers 112 to a content exchange 116 without assistance from content manager software.

An external origin server 118 may have one or more content exchanges 116 assigned to carry content objects for the external origin server 118. The routing database 234 could redirect subscribing client computers to one or more of these content exchanges 116. If the client computer were allowed to use two or more of these content exchanges 116, a client-side routing analysis would be performed on the two or more content exchanges to allow ranking of the relative QOS between them.

The content objects of an external origin server 118 can be preloaded to a content exchange(s) allocated to provide those content objects. To decrease latency when

a content object is requested for the first time, the active directory 104 can crawl the external origin server 118 to determine the content objects available from that server 118. The available content objects may be added to the crawling directory 124. Once the available content objects are known, the active directory 104 requests each content object from the associated content exchange(s) in order to cause loading of each content object on the associated content exchange(s). In this way, content objects are preloaded on the associated content exchanges.

With reference to FIG. 4A, a block diagram of an embodiment of a content exchange portion 116 of the content distribution system 100 is shown. The content exchange 116 caches content objects requested by client computers 112 under the control of the content manager 312. Included in the content exchange 116 are a tracking system 402, and a content node 406. The tracking system includes a content tracker 404, a health check 426, status information 420, a local content catalog 416, and an origin server database 424 while the content node 406 includes a content controller 408 and a content store 412.

The health check 426 can be either a hardware or software application which provides operational characteristics of an associated content exchange 116. In an embodiment, the health check 426 provides a single indication of content exchange 116 status. The single indication may be a normalized value between zero and one indicating a combination of content exchange characteristics including, for example, a CPU load, a CPU temperature, a number of concurrent connections, and a number of requests a content exchange is facilitating. In an alternate embodiment, the health check 426 could monitor characteristics of a content exchange 116 while running on another content exchange 116, origin server 108 or other location.

The content store 412 holds the content objects available for download to the client computers 112 from that content exchange 116. The name of the origin server 108 providing the content object along with path information and the filename is stored with the content object in the content store 412. Via the Internet 120, the client computers 112 connect to the content store 412 and download the content object file or data stream. As new content objects are added to the content store 412, old content objects are typically removed. The age of a content object relates to the last time a content object was accessed. Some content objects on the store 412 never age such that they stay in the store 412 for a predetermined time. An origin server 108 could arrange for the content exchange 112 to store a content object for a predetermined period of time.

When the client computer 112 requests a content object from the content store 412, the content object may not be currently loaded in the content store 412. The content store 412 notifies the content controller 408 of the unfulfilled request for the content object. The content controller 408 locates missing content objects or portions thereof in other content exchanges 116 or from the content server 308 that originated the content object. The missing content objects are loaded into the content store 412 by the content controller 408 such that the client computer 112 can download this information.

When a content object is missing from content store 412, the content controller 408 first checks with other content exchanges 116 to determine if the object is available. If no content exchange 116 has the desired content object, the content server 308 that originated the information is queried for the content object. The content store 412 does not include the IP address for the originating content server 308 so the dynamic DNS 204 is queried for that information. Given the origin server name, the dynamic DNS 204 provides the IP address such that the content controller 408 can request the content object from the proper content server 308.

The content tracker 404 reports to the system 100 the current items in the content store 412 and status information for the content exchange 116. The local content catalog 416 records the origin server name, path and filename for each content object or portion of a content object in the content store 412. As new items are added to and old items are removed from the content store 412, the local content catalog 416 is updated. When a content manager 312 connects to the system 100, a query is made to all content trackers 404 to determine what portions of content objects are stored on the content stores 412. The initial query provides a baseline to which the content tracker 404 updates as changes are made to the various content stores 412. The changes are sent directly to each of the content managers 312 that has content stored in the content store 412. The dynamic DNS 204 is used during this process to determine the IP address corresponding to the origin server name for each content object.

The content tracker 404 also provides status information 420 for the content exchange 116 to the content mangers 312. The status information 420 is sent periodically to each of the content managers 312 as a broadcast or multicast, for example, every five minutes and/or when changes occur. The status information 420 could include the number of concurrent links to the content exchange currently in use, the total number of concurrent links allowed, the bandwidth utilization, the cache churn rate, and/or other information. In other embodiments, the status information is posted to a central location

that the content manager 312 can query when determining where to send a client computer 112 for a downloading a content object.

The content tracker 404 maintains the origin server database 424 to track the origin servers 108 active in the system 100. After attaching to the system, all origin servers 108 identify themselves to the content trackers 404. The content trackers 404 record the origin server name and IP address in the origin server database 424. A query to the dynamic DNS provides the IP address for a given origin server name. If an origin server 108 notifies the content tracker 404 of impending unavailability or if the content tracker 404 cannot contact a particular origin server, the entry for that origin server is removed from the origin server database 424. Additionally, the content corresponding to that origin server 108 may be purged from the content store 412, and the local content catalog 416 is updated. In some embodiments, the content object and content object portions are not purged, but are simply tagged for deletion as the storage space is requested.

In some embodiments, the content controller 408 can be instructed by the system 100 to acquire and retain predetermined content objects in the content store 412. Content objects that are anticipated to be wanted can be preloaded in preparation for the demand. The desirability of a content object can be determined by monitoring click-throughs to those content objects from the search and directory pages 212, 216. For example, content objects related to a famous person could be loaded onto content stores 412 immediately before a biography program on the famous person is aired on network television. Alternatively, users could subscribe to a service that loads content objects to some content exchanges 116. For example, a scheduled network program could be loaded to coincide with a TV broadcast the user could view over the Internet 120 in lieu of the TV broadcast. In this way, the content would be available without delay to the subscribed users.

With reference to FIG. 4B, a block diagram of another embodiment of a content exchange portion 116 of the content distribution system 100 is shown. This embodiment includes multiple content nodes 406 coupled to a single tracking server 402. A content bus 428 allows content nodes 406 to check each others content stores 412 for missing content objects. The content bus 428 could also couple to other content notes in other locations. The content bus 428 may or may not travel in part over the Internet 120.

Referring next to FIG. 4C, a block diagram of an embodiment of a content exchange site 432 including multiple content exchange servers 116 is shown. Even

though the content exchange site 432 includes multiple content exchange servers 116, it appears to the system a single content exchange 116. The load of the content exchange site 432 is distributed among the constituent content exchange servers 116. A switch 436, such as a layer four switch, distributes the content object requests to the content exchanges 116 and aggregates the spooled responses to the Internet 120.

Referring next to FIG. 5, a block diagram of an embodiment of a client computer portion 112 of the content distribution system 100 is shown. The client computer 112 communicates to the Internet 120 in order to deliver content to a user. Included in the client computer 112 are a viewer object proxy 504, a content processing program 508, preference information 512, a network interface 516, and hosted server routing 520.

The content processing program 508 is typically software that interprets or processes a content object downloaded from the Internet 120. Examples of content processing programs 508 include web browsers, file transfer protocol (FTP) software, gopher software, news (NNTP), mail programs, streaming media players, non-streaming media players, and other software. The Internet communication from content processing program 508 that is normally sent directly to the Internet is redirected to the viewer object proxy 504.

The viewer object proxy 504 serves as intermediary between the Internet 120 and the content processing program 508. After installation of the viewer object proxy 504, it determines its general location relative to known points on the Internet 120. Content exchanges 116 that are reasonable candidates for providing sufficient QOS are tested to determine the number of hops necessary to reach each content exchange 116 and the latencies between each content exchange 116 and the viewer object proxy 504. A weighting of QOS factors, such as the number of hops and bandwidth achieved, is recorded as preference information 512 and is passed as meta-data to the content manager 312 in an HTTP header.

Other embodiments could pass meta-data in any sort of data channel and not just through a HTTP header. For example, the meta-data could travel through a dedicated port, an IP address, a URL, a header, or other logical channel.

The preference information 512 is the result of network analysis performed from the client computer perspective 112. When a content object is requested, the preference information 512 is communicated to the content object manager 312 which in turn selects an appropriate content exchange 116 for the client computer 112.

Periodically, such as every hour, the preference information 512 is updated using automated tests or is updated manually by the user. Subsequent tests take into account the previous results to efficiently consider preferred content exchanges. For example, the first analysis may check one hundred content exchanges, but a subsequent analysis could eliminate the poor performing content exchanges such that only fifty are analyzed.

The preference information 512 includes a list of content exchanges 116 and their associated QOS values resulting from client-side network analysis. In some embodiments, there are multiple pathways to an external origin server 118. The multiple pathways are separated by port, IP address, server identification (ID), and/or other mechanisms. Client-side network analysis could be used to determine a QOS value associated with each pathway to an external origin server 118 or any other source of a content object that could also have multiple pathways.

After the user of the client computer 112 chooses a content object, the origin server name is provided to the viewer object proxy 504. The origin server name is used for the viewer object proxy 504 to query the dynamic DNS 204 for the IP address of the origin server 108. Once the IP address is known, the content processing program 508 is redirected to the content manager 312 for the desired content object. The content manager 312 is passed the preference information 512 to allow routing the client computer 112 to the appropriate content exchange 116. In this embodiment, the preference information 512 includes the preferred ten content exchanges, but the amount of preferred content exchanges could be adjusted by the user.

In some embodiments, the viewer object proxy 504 can be HTTP-specific but protocol independent for routing information. Thus, routing information is transferred according to HTTP, but the actual routing information is protocol independent. However, one skilled in the art would recognize that the viewer object proxy 504 can also be configured to work with other network protocols as needed. For example, the viewer object proxy 504 can also be configured to operate in accordance with FTP, NNTP, RTP, RTSP, SMTP, or SHOUT etc.

The client computer 112 includes a host server routing database 520. This host server routing database 520 can include information related to external origin servers 118 accessible by the client computer 112. In an embodiment, host server routing database 520 is a subset of the entries in the routing database 234 found in the active directory 104.

The client computer 112 includes a network interface 516 that connects the viewer object proxy 504 to the Internet 120. Common examples of network interfaces 516 include analog modems, DSL modems, ISDN, cable modems, satellite modems, cellular modems, power line modems, line-of-sight microwave modems, line-of-sight laser modems, etc.

In this embodiment, the client computer is associated with a home user. In other embodiments, the client computer could serve digital movies to a theater or provide content objects to a corporate network user, a hotel patron or apartment complex.

With reference to FIG. 6, a block diagram of an embodiment of a content distribution system 600 is shown. This figure depicts data flow between data blocks without showing transport over the Internet 120 or any other network. It is to be understood, however, that the Internet 120 or another packet switched network is used in some embodiments. Also, the figure is somewhat simplified in that some blocks from Figs. 2-5 and the external origin server(s) 118 and crawling directory 124 are not included to simplify the FIG. 6.

The network interface 516 is the connection to the Internet 120 for the client computer 112. The client computer 112 connects to the directory and search pages 212, 216 to allow the user to select content objects for download. When a content object is selected, redirection of the client computer 112 from the active directory 104 to the appropriate origin server 108 uses the dynamic DNS 204. Preference information 512 is passed to the content manager 312 to assist its selection of the source for the content object. Depending on the selection made by the content manager 312, the content object is downloaded from one of the content exchanges 116 or from the content server 308.

The active directory 104 interacts with the other modules in the system 600. The client computer 112 accesses the directory and search pages 212, 216 to select a content object. The content tracker 404 and content manager 312 respectively provide status and catalog information to the server manager 208. Account information is provided to the server manager 208 by the administrator of the origin server 108 and by the user of client computer 112 to maintain the subscriber database 224. Redirection from the origin server name to the IP address of the origin server 108 is provided to the viewer object proxy 504, the content tracker 404 and the content controller 408 by the dynamic DNS 204.

The origin server 108 communicates with the server manager 208, the client computer 112, the content tracker 404, the content store 412, and the content

controller 408. The local content catalog 316 is provided to the server manager 208 from the content manager 312 in order to maintain the server database 228 with current content information. Preference information 512 is provided to the content manager 312 from the client computer 112 to facilitate selection of a source of the content object. The content tracker 404 interacts with the content manager 312 to know what content objects are stored on the content exchange 116. Content objects are read from the content server 308 by either the content store 412 or the client computer 112.

The content exchanges 116 interact with the other modules in the system 600 as well. Status information is provided to the active directory 104 and/or the content manager 312. The dynamic DNS is used by both the content controller 408 and the content tracker 404 to find the IP address of an origin server 108 that contains a content object. A selected content exchange may contact other content exchanges when an object is needed for the content store 412. If the other content exchanges do not have the content object, the content controller 408 requests the object from the content server 308 for delivery to the selected content store 412.

Referring next to FIG. 7, a block diagram of another embodiment of the content distribution system 700 is shown that has two types of content exchanges 702, 704. This embodiment includes public content exchanges 702 and captive content exchanges 704. One or more associated client computers 708 and one or more associated origin servers 712 are grouped with the one or more captive content exchanges 704. The public content exchanges 702 are available to source content objects to any of the client computers 112 or any of the associated client computers 708. Those content objects stored on the public content exchanges 702 could originate from any of the unassociated origin servers 106, origin servers 108, external origin servers 118, or associated origin servers 712.

The captive content exchanges 704, however, are not fully available to the system 700 in the same way as the public content exchanges 702 are. The captive content exchanges 704 cache content objects originating from associated origin servers 712. Those cached objects are provided to either the associated client computers 708 or client computers 112. Content objects originating from the unassociated origin servers 106, origin servers 108 and external origin servers 118 are cached for associated client computer 708, but not for client computers 112. The client computers 112 cannot rely on the captive content exchanges 704 for caching content objects that were sourced from any unassociated origin servers 106, origin servers 108 or external origin servers 118.

Accordingly, the preference information 512 for the client computer 112 only includes public content exchanges 702 and not captive content exchanges 704.

The captive content exchanges 704 in this embodiment are linked to the public content exchanges 702 such that missing content objects or portions of content objects may be passed between content exchanges 702, 704 when requested by a user. When a content object is requested from the preferred content exchange 702, 704 a local search is first performed for the content object or any portion of the content object. If that search fails, the other content exchanges 702, 704 are searched before resorting to requesting the content object from the origin server 106, 108, 118, 712 that first provided the content object to the system 700. Other embodiments, however, could go without passing files between the captive content exchanges 704 and public content exchanges 702.

In various embodiments, the captive content exchanges 704 could be located in the central office, headend, ISP location, or nodes where the Internet bandwidth begins to travel the "last mile" or final leg to the associated client computer 708. By moving the cache as close as possible to the destination, the potential bottlenecks are reduced to improve QOS. For example, a cable multi-system operator (MSO) providing service to associated client computers 708 by way of DOCSIS modems could have captive content exchanges 704 in the headend or in neighborhood nodes. In this way, the associated client computers 708 and associated origin servers 712 connect to the captive content exchange 704 to provide enhanced QOS through caching.

With reference to FIG. 8, a block diagram of yet another embodiment of the content distribution system 800 is shown. This embodiment demonstrates how the captive content exchange 704, associated client computers 708 and associated origin servers 712 can be virtually connected through the Internet 120. These functional blocks 704, 708, 712 could be coupled in any way. For example, the captive content exchanges 704 could be coupled to the associated client computers 708 through a hybrid fiber/cable distribution system. But, the associated origin servers 712 could be linked to the captive content exchanges 704 through the Internet 120.

Referring next to FIG. 9, a flow diagram of an embodiment of a process 900 for populating captive content exchanges 704 that has available storage capacity is shown. The depicted process 900 picks-up in step 904 where a test detects bandwidth utilization below a threshold. The bandwidth measurement for this test could be the bandwidth from all associated origin servers 712 to the captive content exchange 704 in

this embodiment or could be the bandwidth from a single associated origin server 712 to the captive content exchange 704 in other embodiments.

In step 908, a determination of which origin servers are associated origin servers 712 is made. The active directory 104 could be queried for this information or the information could be maintained local to the captive content exchange 704. Once the associated origin servers 712 are known, each local content catalog 316 is queried to determine the content objects available on that associated origin server 712 in step 912.

In step 916, the content objects from all the associated origin servers 712 are analyzed to determine a ranking between the content objects. In this embodiment, the popularity of the content objects controls the ranking. The number of times a content object is downloaded over the last day is the algorithm used for determining popularity, however, other embodiments could use other time periods. Although this embodiment uses popularity for ranking, other embodiments could also use size of the content object, price of the content object, and/or other factors when determining the ranking.

The associated origin servers 712 or captive content exchanges 704 could monitor popularity by respectively determining the number of click-throughs or hits for a content object. Click-throughs of the user of the client computer can be monitored with CGI scripts that redirect a content object selection back to the associated origin server 712. Once redirected back, the click-through is recorded and the client computer 112, 708 is redirected to the content exchange 702, 704 that serves as the source of the content object.

In step 920, a determination of how much cache space in the content store 412 will be allocated to storing the popular content objects. This embodiment uses half of the content store 412 to store popular content, but other embodiments could use other fractions of the capacity of the content store 412.

A determination is made in step 922 as to whether any of the content objects are stored in the content store 412 already. To determine what is already stored, the local content catalog 316 for the associated origin server 712 is cross referenced against the local content catalog 416 for the captive content exchange 704 to determine which of the content objects are not already stored in step 922. Those content objects that are already stored are marked as not needing any reload in step 924.

The highest ranked content objects that will fit into the allocated cache space and are not already loaded are determined in step 928. Those content objects that will not fit are removed from the list. Starting with the least popular on the ranked list,

the content objects that are not already in the content store 412 are requested such that they are loaded onto the content store 412. The content objects that are already loaded on the content store 412 are moved to the front of the cache when their entry is reached when cycling through the list. In this way, all the popular content objects that will fit are loaded in the cache according to ranking. Those with lower rankings will be pushed out of the back of the cache first if there are no subsequent hits or requests for those content objects.

In some embodiments, each associated origin server 712 is harvested for content objects in a separate process that is run in parallel to other processes harvesting other associated origin servers 712. The bandwidth between a specific associated origin server 712 and the captive content exchange 704 is tested to determine if bandwidth utilization has fallen below a threshold. Once the threshold is crossed, the content objects are loaded onto the captive content exchange. In parallel, other associated origin servers 712 are harvested for content objects.

With reference to FIG. 10, a flow diagram of another embodiment of a process 1000 for populating the captive content exchange 704 is shown. In this embodiment, a file system rather than a cache is used. Content objects loaded into the file system remain until an expiration time. For example, a content object could be loaded one day with an expiration period of one day such that only twenty-four hours later can the file be overwritten.

The notable differences between this embodiment and the embodiment of FIG. 9 begin in step 1004 where the space in the file system reserved for preloading is determined. The file system space is logically separated from the space used for cacheing. Content objects that are stored on the file system should not be redundantly cached. In step 1002 and 924, entries for files that are already stored in the cache are marked such that they are not reloaded. The content objects that will not fit on the file system are removed from the list in step 1008. Any content objects that are not already loaded and that will fit are loaded in step 1012. The expiration time for all the content objects on the file system is set for one day or some other time period.

With reference to FIG. 11, a flow diagram of yet another embodiment of a process 1100 for populating the captive content exchange 704 is shown. In this embodiment, the captive content exchange 704 populates itself after becoming available to the system 700. For example, the captive content exchange 704 may populate itself after powering on. This embodiment does not determine which content objects are

already stored in the cache. Instead, all content objects that are determined to fit are loaded.

The above embodiments in FIGS. 9-11 populate captive content exchanges 704. Other embodiments could also populate public content exchanges 702. A process in the active directory 104 would determine the popular content objects on the Internet and where those objects are likely to be requested from the distributed network of public content exchanges 702. The active directory could instruct public content exchanges 702 to preload content such that it is available with enhanced QOS.

Referring next to FIG. 12, a flow diagram of an embodiment of a process 1100 for populating a distributed storage system with content from an origin server 106, 108, 118, 712 is shown. In this embodiment, the origin server 712 populates the captive content exchange 704 when it becomes available to the system 700 if hosting is allowed. The depicted process picks-up in step 1204 as the origin server 106, 108, 118, 712 comes online. In step 1208, a determination is made as to whether there is a captive content exchange 704 associated with the origin server 106, 108, 118, 712. Where there is no captive content exchange 704 no preloading occurs and processing continues to step 1212 and 1240 where the request for a content object is fulfilled by the preferred content exchange 702, 704.

In step 1220, a further determination is made as to whether hosting of the content objects on the captive content exchange 704 is allowed for this origin server 106, 108, 118, 712. A captive content exchange 704 may require compensation for the ability of an origin server 106, 108, 118, 712 to offload its content objects. If hosting is allowed, the popularity of the content objects is determined in step 916. In step 1228, the content objects are loaded onto the captive content exchange 704 such that the most popular objects are last to be expired out the back of the cache. Other embodiments could use a file system, with lifetime information specified for each content object as described above.

If hosting is not allowed in step 1220, conservation of upstream bandwidth is possible by first populating the captive content exchange 704 with the content object. Once the captive content exchange 704 has the content object it can propagate to other content exchanges 702, 704 efficiently. In step 1224, the client computer 112 requests the content from the origin server 106, 108, 118, 712. A determination in step 1232 is performed as to whether the content is already stored in any content exchange 702, 704. If the content is already stored in a content exchange 702, 704 away from the origin

server 106, 108, 118, 712, the client computer 112 is directed to the preferred content exchange 702, 704 in step 1240. If needed, the preferred content exchange 702, 704 can get the content object from any other content exchange 702, 704 that already has the content object.

If the content object is not available on the distributed storage network of content exchanges 702, 704 as determined in step 1232, the client computer 112 is directed first to the captive content exchange 704 associated with the origin server 712. In this way, the captive content exchange 704 preferentially stores content objects from associated origin servers 712.

With reference to FIG. 13, a flow diagram of an embodiment of a process 1300 for providing a list of content exchanges 702, 704 to a client computer 112, 712 is shown. In this embodiment, a check for captive content exchanges 704 is performed before providing a list of preferred content exchanges 702, 704 to the client computer 708. The depiction of the process picks-up in step 1304 where a determination of which captive content exchanges 704 are associated with which associated client computers 708. These associations are recorded in a central location such as the active directory 104.

In step 1308, a client computer 112, 708 requests a list content exchanges 702, 704 from the active directory 104. A determination is made as to whether there are any captive content exchanges 704 in step 1312. If there are captive content exchanges 704, processing continues to step 1308 where those captive content exchanges 704 are included in the list. In step 1312, other public content exchanges 702 likely to have sufficient or adequate QOS are added to the list. If there are no captive content exchanges 704 as determined in step 1312, step 1308 is skipped and the list of preferred content exchanges 702 is assembled without any captive content exchanges 704.

Once the list of preferred content exchanges 702, 704 is assembled, it is sent to the client computer 112 in step 1316 and stored in the preference information database 512. The client computer 112 analyzes each content exchange 702, 704 on the list to rank them. Typically, the captive content exchanges 704 can provide the best QOS to associated client computers 708.

In light of the above description, a number of advantages of the present invention are readily apparent. For example, QOS is improved by preloading content objects onto a content exchange. By preloading based upon triggering events, the content object is passed through the upstream connection between the associated origin server and the captive content exchange in a controlled manner. Once a content object is offloaded

to the captive content exchange, the associated origin server is likely to be less taxed by requests for the content object.

A number of variations and modifications of the invention can also be used. For example, some of the above embodiments detect low bandwidth utilization to trigger preloading of the content exchange. But, other embodiments could preload at a preset time, such as the middle of the night. Presumably, bandwidth utilization is at a minimum in the middle of the night.

While the principles of the invention have been described above in connection with specific apparatuses and methods, it is to be clearly understood that this description is made only by way of example and not as limitation on the scope of the invention.

WHAT IS CLAIMED IS:

1. A method for loading content objects in a content store on a network, the method comprising steps of:

waiting for a triggering event;

determining if a first origin server is authorized to store content in the content store;

loading a first content object from the first origin server onto the content store without a request for the first content object;

determining if a second origin server is authorized to store content in the content store; and

loading a second content object from the second origin server onto the content store without a request for the second content object.

2. The method for loading content objects in the content store on the network as recited in claim 1, further comprising a step of determining the popularity of the content object.

3. The method for loading content objects in the content store on the network as recited in claim 1, wherein the performance of the loading steps are conditioned on the waiting step.

4. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises at least one of the following steps of waiting for:

upstream bandwidth between the first origin server and the content store to fall below a predetermined threshold; and

upstream bandwidth between the second origin server and the content store to fall below a predetermined threshold.

5. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises a step of waiting for upstream bandwidth into the content store to fall below a predetermined threshold.

6. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises a step of waiting for connection to the network.

7. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises a step of waiting for upstream bandwidth utilization from a client computer to the content store to fall below a predetermined threshold.

8. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises a step of waiting for a temporal event.

9. A method for loading content objects in a content store on a network, the method comprising steps of:

waiting for a triggering event;

determining the popularity of the content object;

loading a first content object from the first origin server onto the content store without a request for the first content object; and

loading a second content object from the second origin server onto the content store without a request for the second content object.

10. The method for loading content objects in the content store on the network as recited in claim 1, wherein the performance of the loading steps are conditioned on the waiting step.

11. The method for loading content objects in the content store on the network as recited in claim 1, further comprising steps of:

determining if a first origin server is authorized to store content in the content store; and

determining if a second origin server is authorized to store content in the content store.

12. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises at least one of the following steps of waiting for:

upstream bandwidth between the first origin server and the content store to fall below a predetermined threshold; and

upstream bandwidth between the second origin server and the content store to fall below a predetermined threshold.

13. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises a step of waiting for upstream bandwidth into the content store to fall below a predetermined threshold.

14. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises a step of waiting for connection to the network.

15. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises a step of waiting for upstream bandwidth utilization from a client computer to the content store to fall below a predetermined threshold.

16. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises a step of waiting for a temporal event.

17. A method for loading content objects in a content store on a network, the method comprising steps of:

waiting for a triggering event;

loading a first plurality of content objects from the first origin server onto the content store without a request for any of the first plurality of content objects; and

loading a second plurality of content objects from the second origin server onto the content store without a request for the second plurality of content objects, wherein the performance of the loading steps are conditioned on the waiting step.

18. The method for loading content objects in the content store on the network as recited in claim 1, further comprising a step of determining the popularity of the content object.

19. The method for loading content objects in the content store on the network as recited in claim 1, further comprising steps of:

determining if a first origin server is authorized to store content in the content store; and

determining if a second origin server is authorized to store content in the content store.

20. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises at least one of the following steps of waiting for:

upstream bandwidth between the first origin server and the content store to fall below a predetermined threshold; and

upstream bandwidth between the second origin server and the content store to fall below a predetermined threshold.

21. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises a step of waiting for upstream bandwidth into the content store to fall below a predetermined threshold.

22. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises a step of waiting for connection to the network.

23. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises a step of waiting for upstream bandwidth utilization from a client computer to the content store to fall below a predetermined threshold.

24. The method for loading content objects in the content store on the network as recited in claim 1, wherein the waiting step comprises a step of waiting for a temporal event.

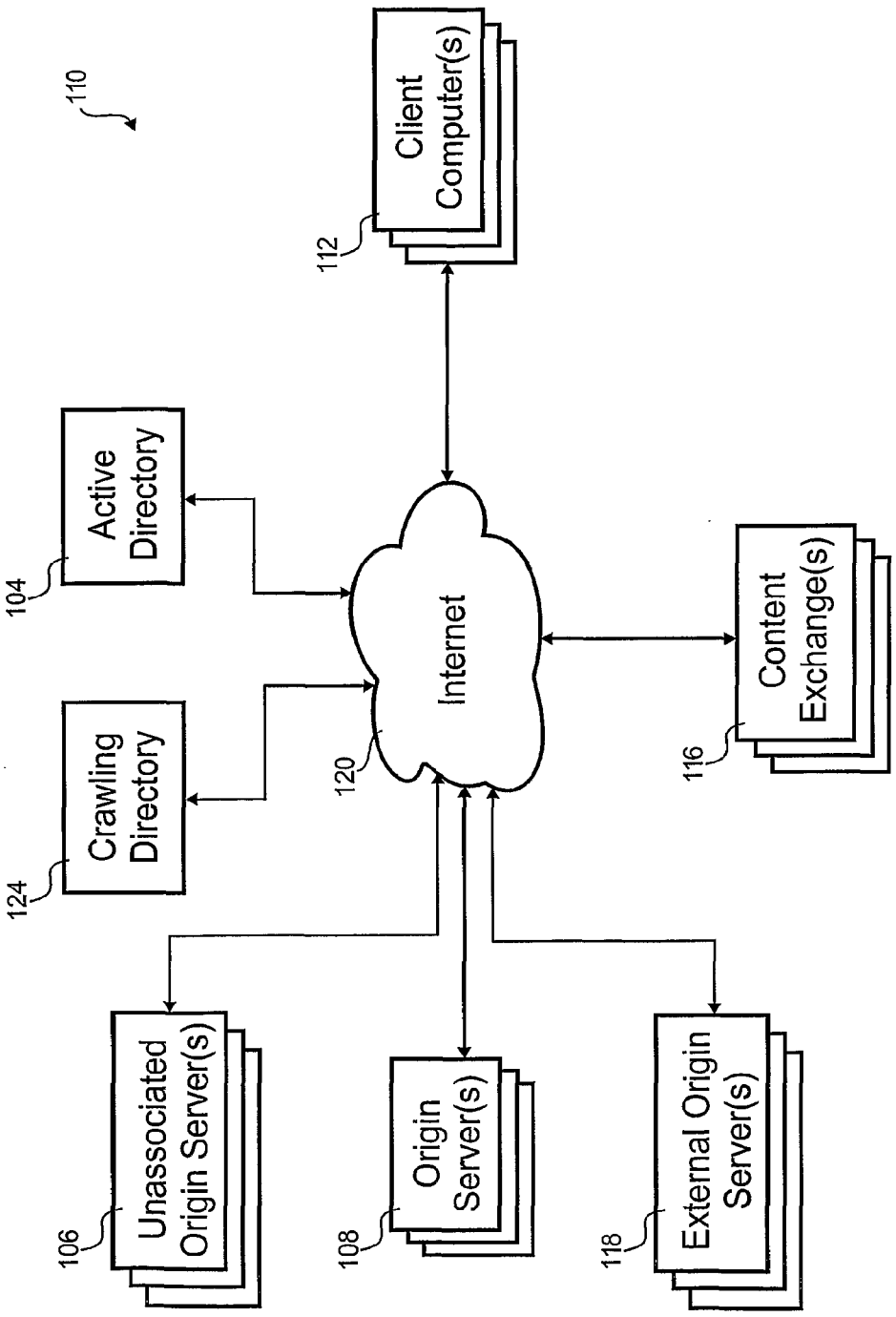

110
Client Computer(s)
112
Active Directory
104
Internet
120
Content Exchange(s)
116
Crawling Directory
124
Unassociated Origin Server(s)
106
Origin Server(s)
108
External Origin Server(s)
118

Fig. 1

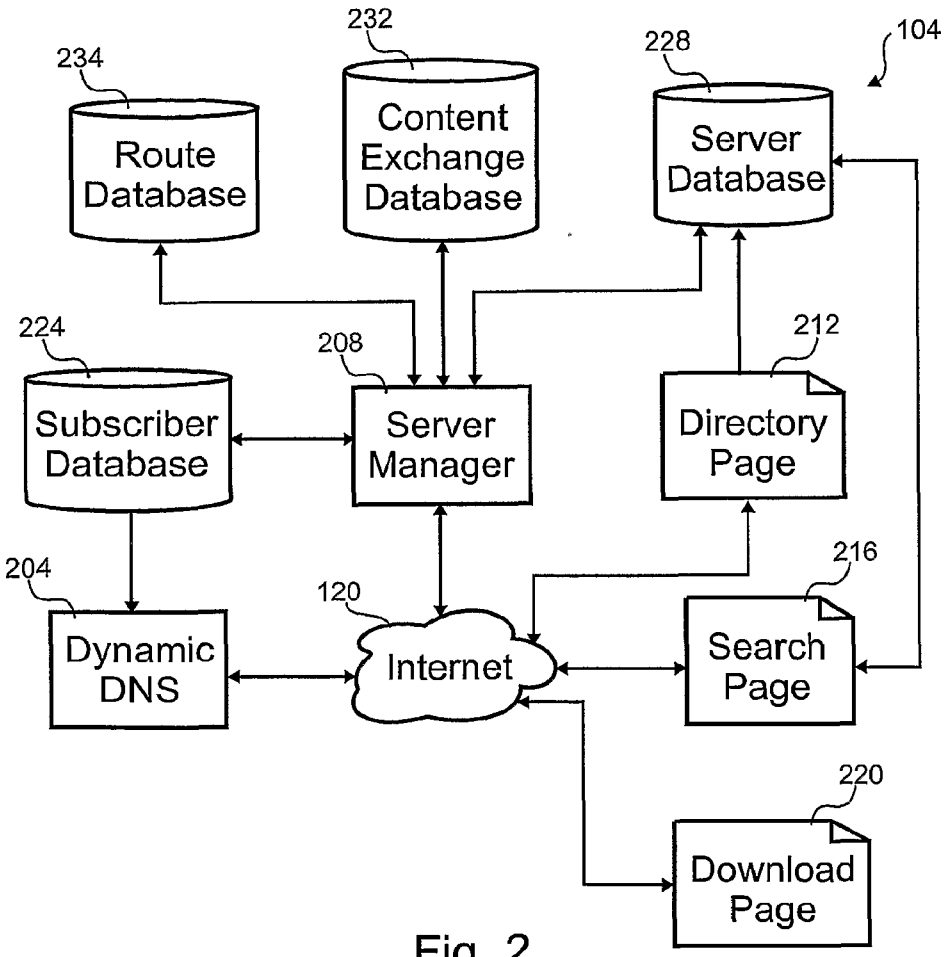

232
234
228
104
Route Database
Content Exchange Database
Server Database
224
208
212
Subscriber Database
Server Manager
Directory Page
204
120
216
Dynamic DNS
Internet
Search Page
220
Download Page

Fig. 2

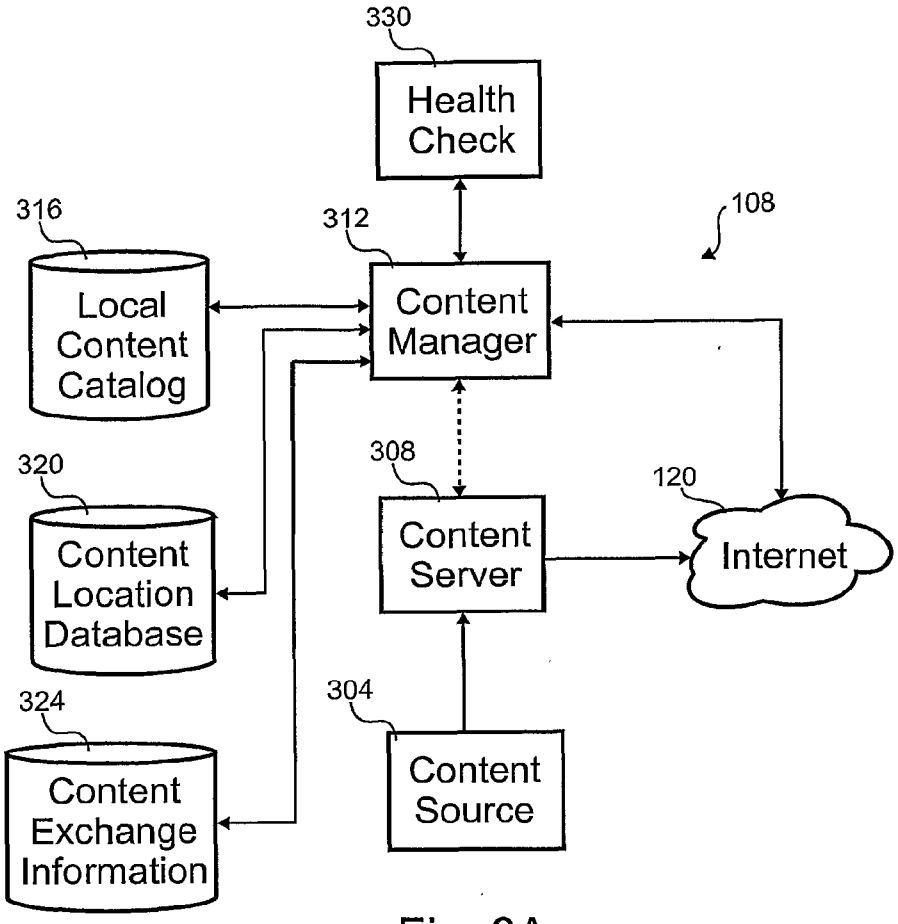

330
Health Check
316
312
108
Local Content Catalog
Content Manager
320
308
120
Content Location Database
Content Server
Internet
324
304
Content Exchange Information
Content Source

Fig. 3A

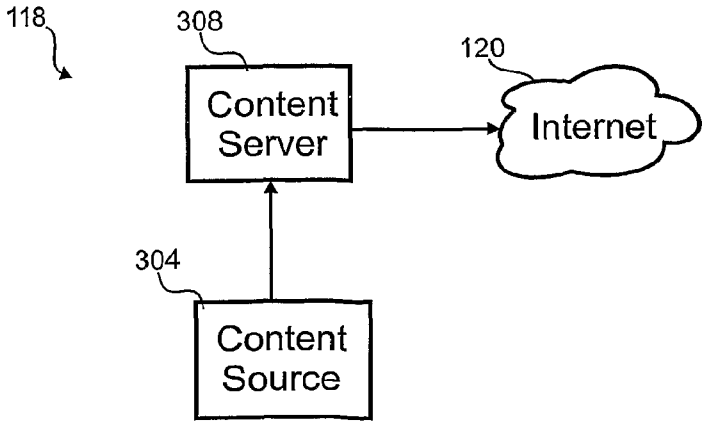

118
308
120
Content Server
Internet
304
Content Source

Fig. 3B

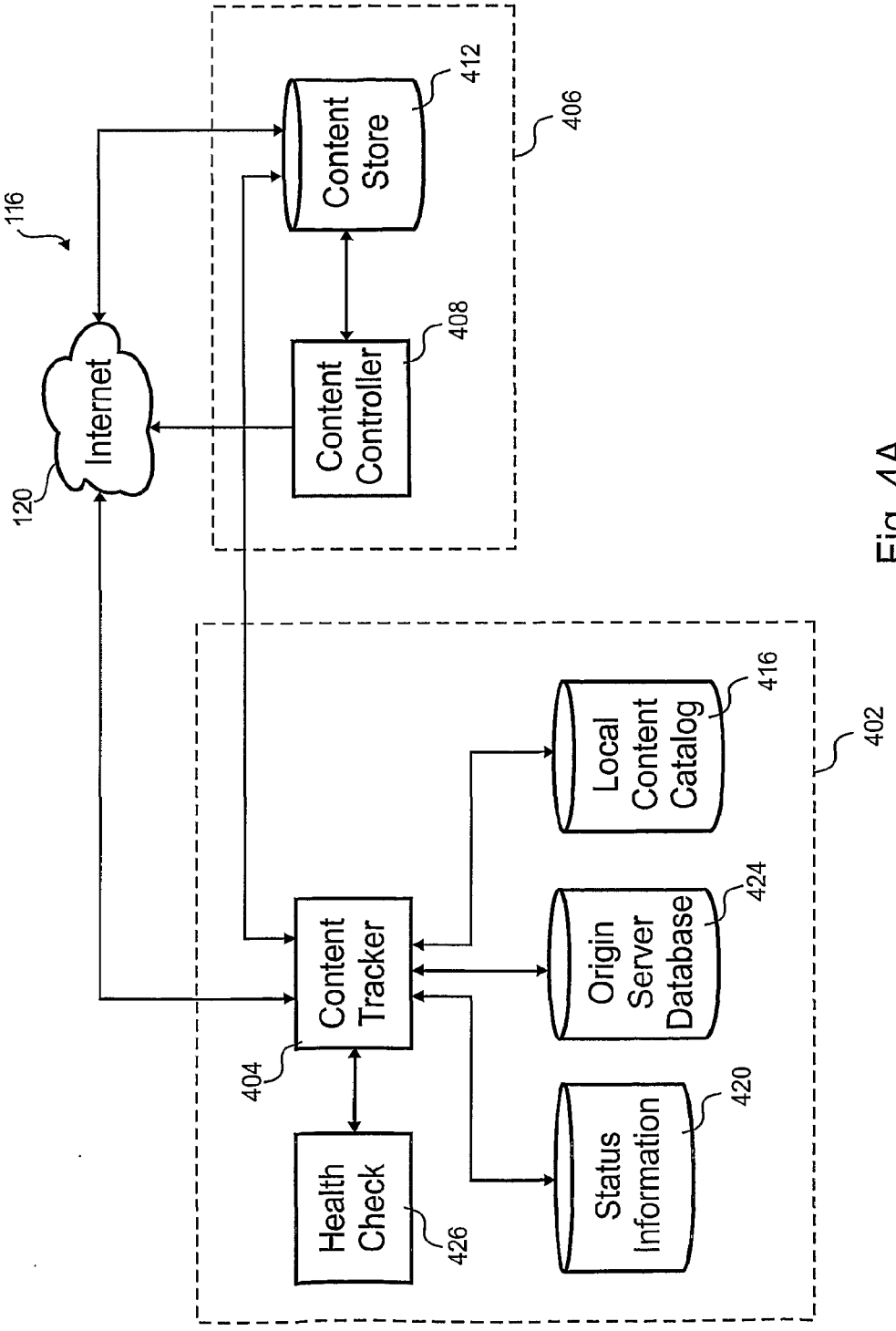

116
Internet
120
Content Store
412
406
Content Controller
408
Content Tracker
404
Health Check
426
Local Content Catalog
416
Origin Server Database
424
Status Information
420
402

Fig. 4A

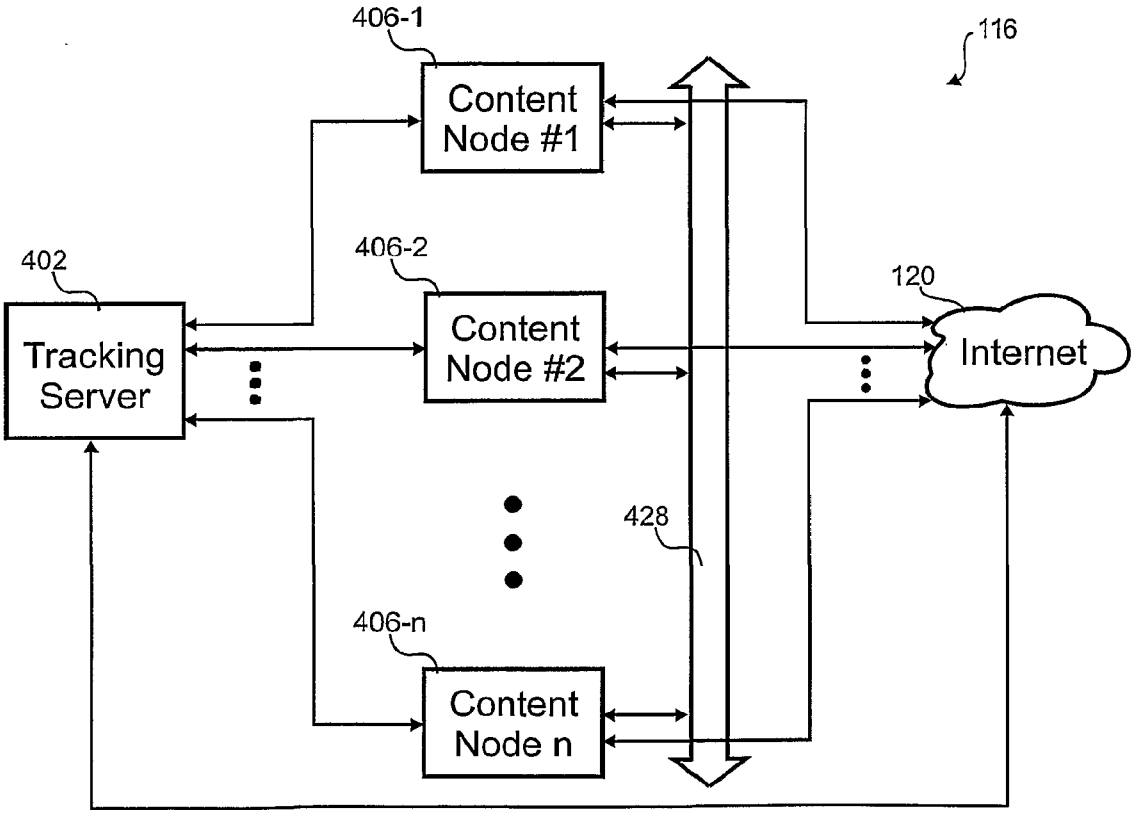

406-1
116
Content Node #1
402
406-2
120
Tracking Server
Content Node #2
Internet
428
406-n
Content Node n

Fig. 4B

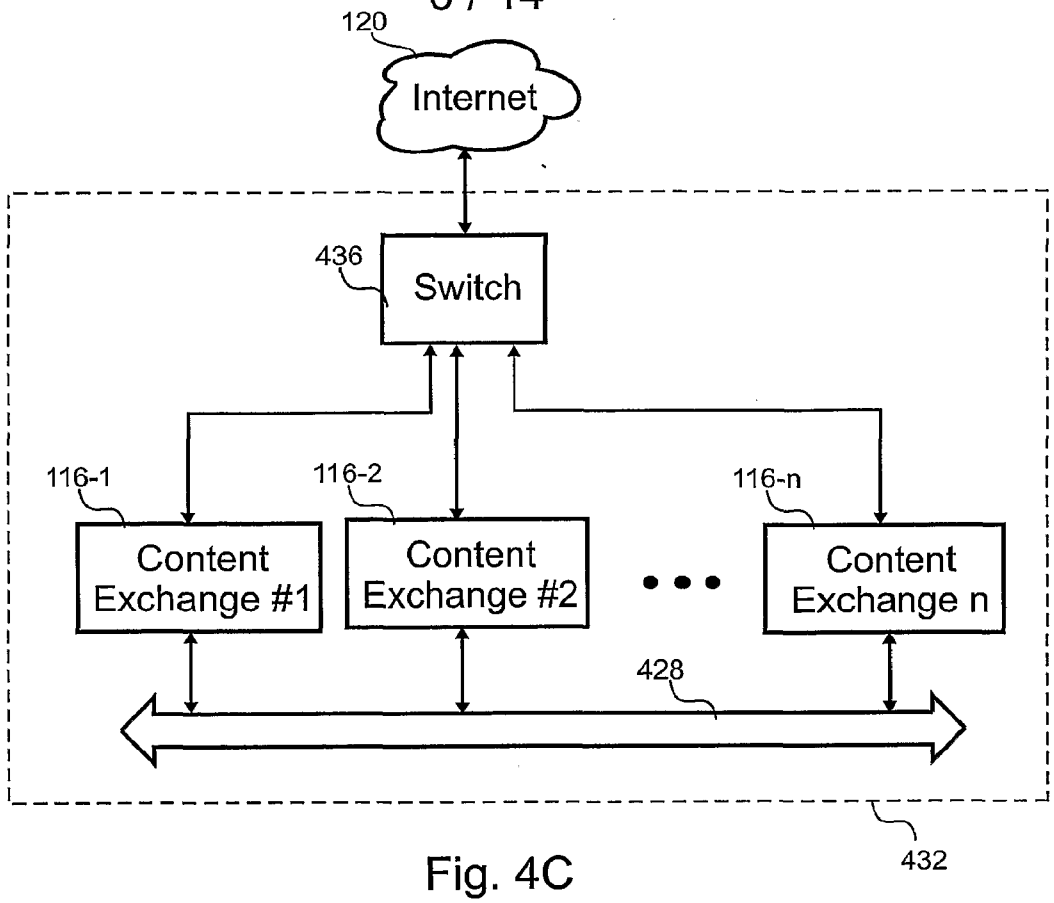

120
Internet
436
Switch
116-1
116-2
116-n
Content Exchange #1
Content Exchange #2
Content Exchange n
428
432

Fig. 4C

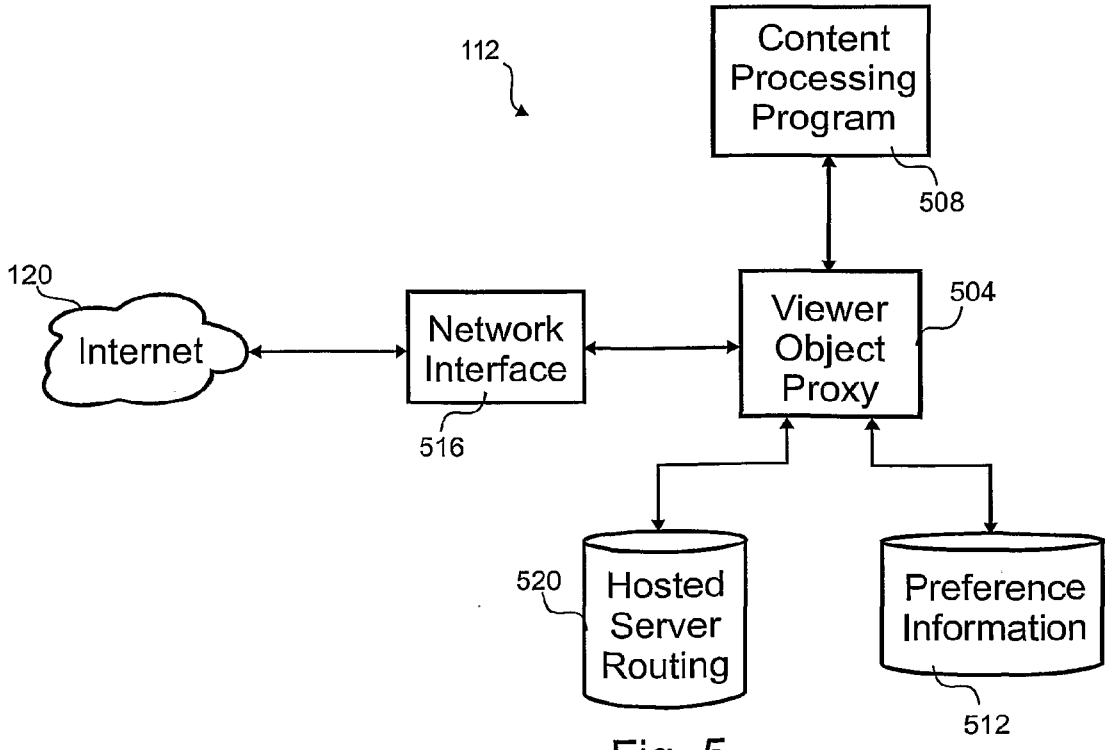

112
Content Processing Program
508
120
Internet
Network Interface
516
Viewer Object Proxy
504
520
Hosted Server Routing
Preference Information
512

Fig. 5

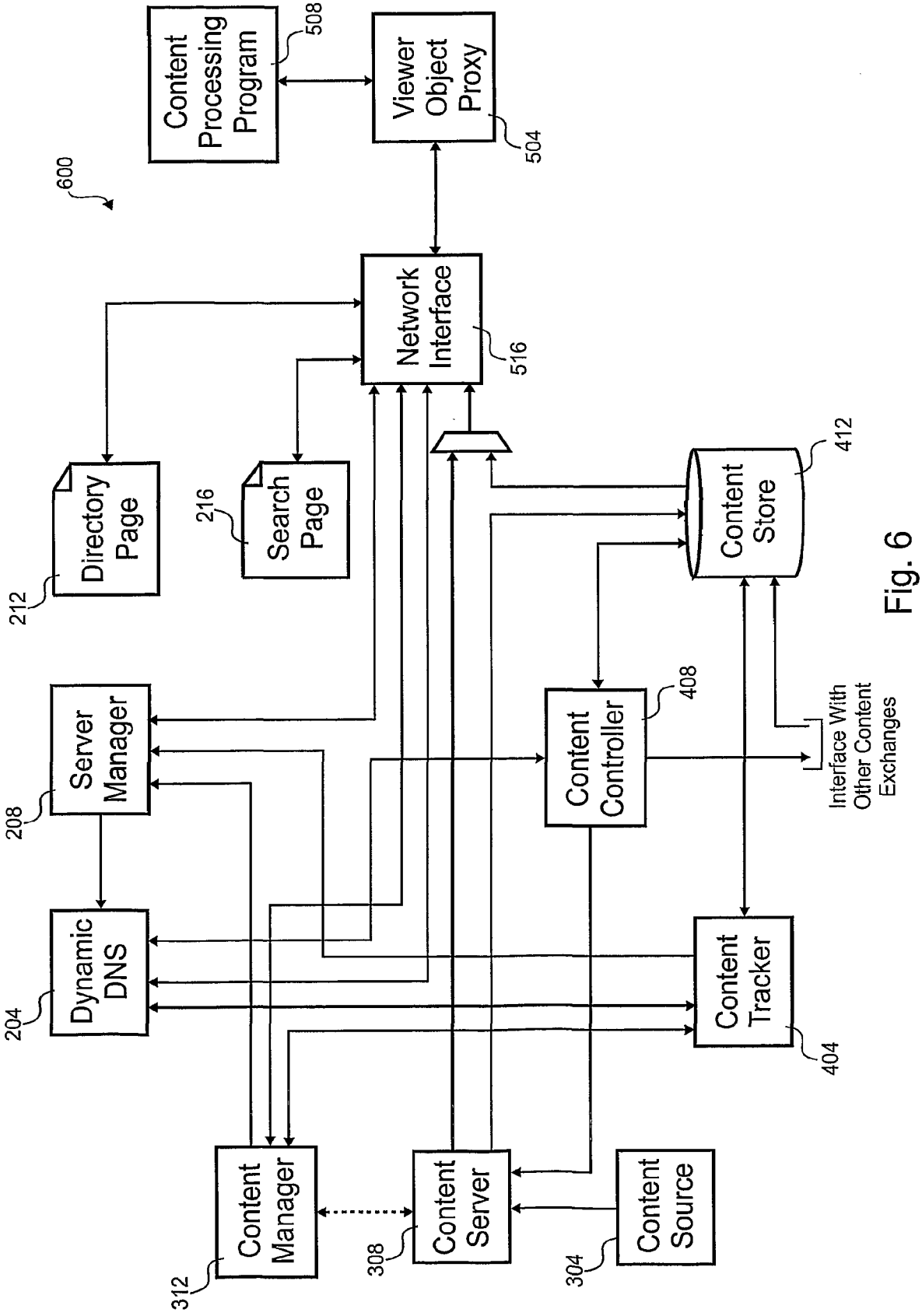

600
Content Processing Program
508
Viewer Object Proxy
504
Network Interface
516
Directory Page
212
Search Page
216
Content Store
412
Server Manager
208
Content Controller
408
Interface With Other Content Exchanges
Dynamic DNS
204
Content Tracker
404
Content Manager
312
Content Server
308
Content Source
304

Fig. 6

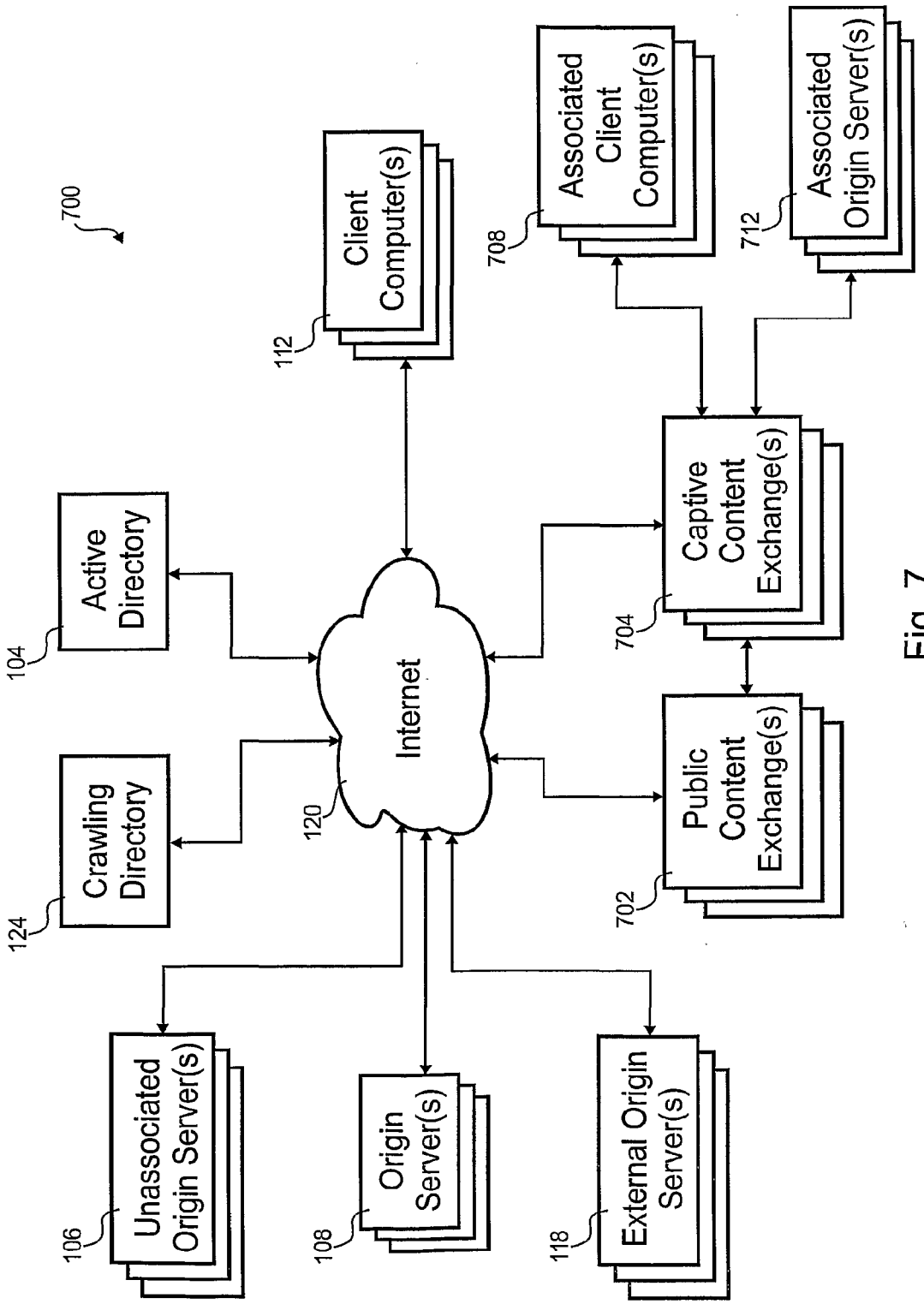

700
Client Computer(s)
112
Associated Client Computer(s)
708
Associated Origin Server(s)
712
Active Directory
104
Internet
120
Captive Content Exchange(s)
704
Public Content Exchange(s)
702
Crawling Directory
124
Unassociated Origin Server(s)
106
Origin Server(s)
108
External Origin Server(s)
118

Fig. 7

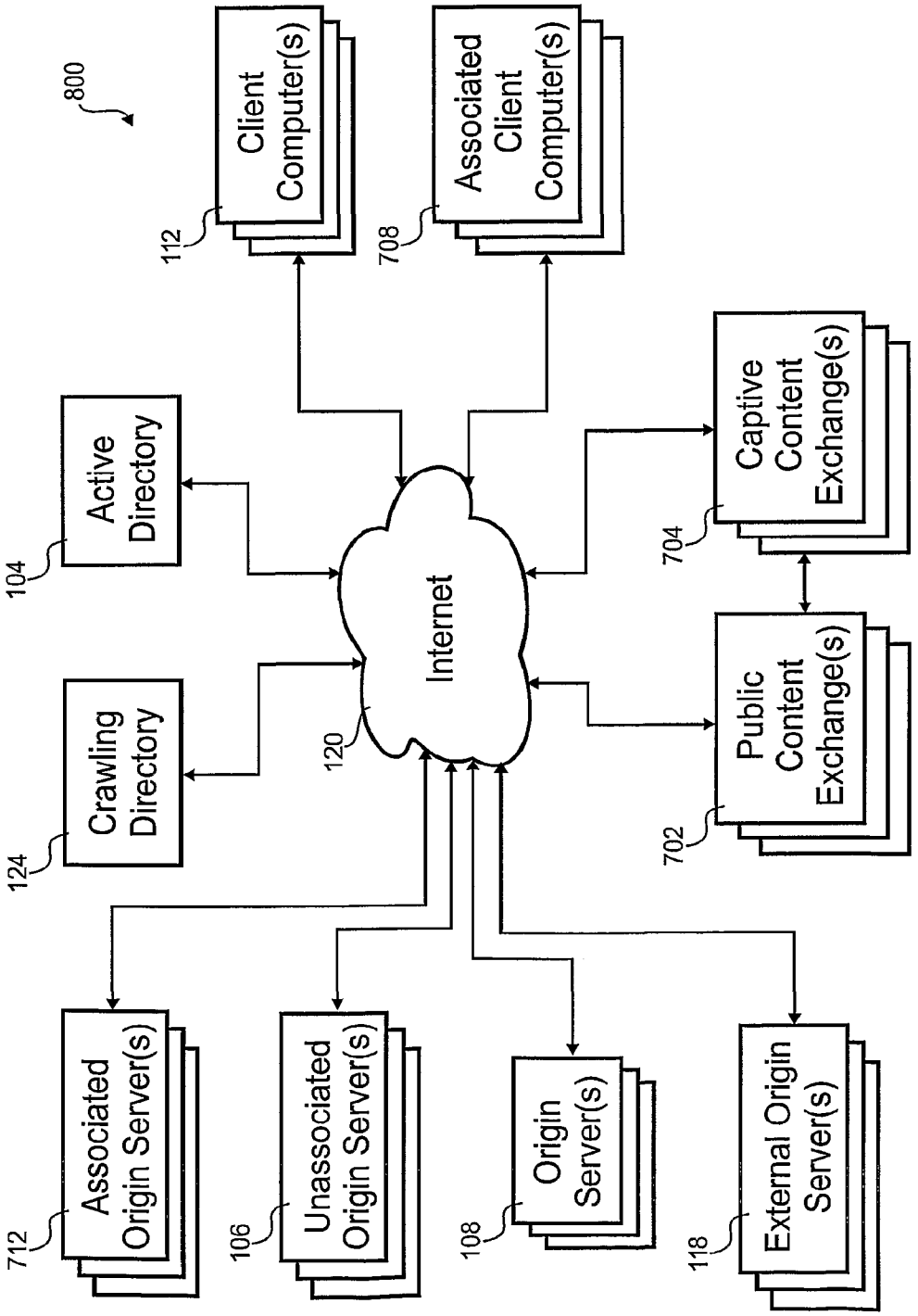

800
Client Computer(s)
112
Associated Client Computer(s)
708
Active Directory
104
Internet
120
Captive Content Exchange(s)
704
Public Content Exchange(s)
702
Crawling Directory
124
Associated Origin Server(s)
712
Unassociated Origin Server(s)
106
Origin Server(s)
108
External Origin Server(s)
118

Fig. 8

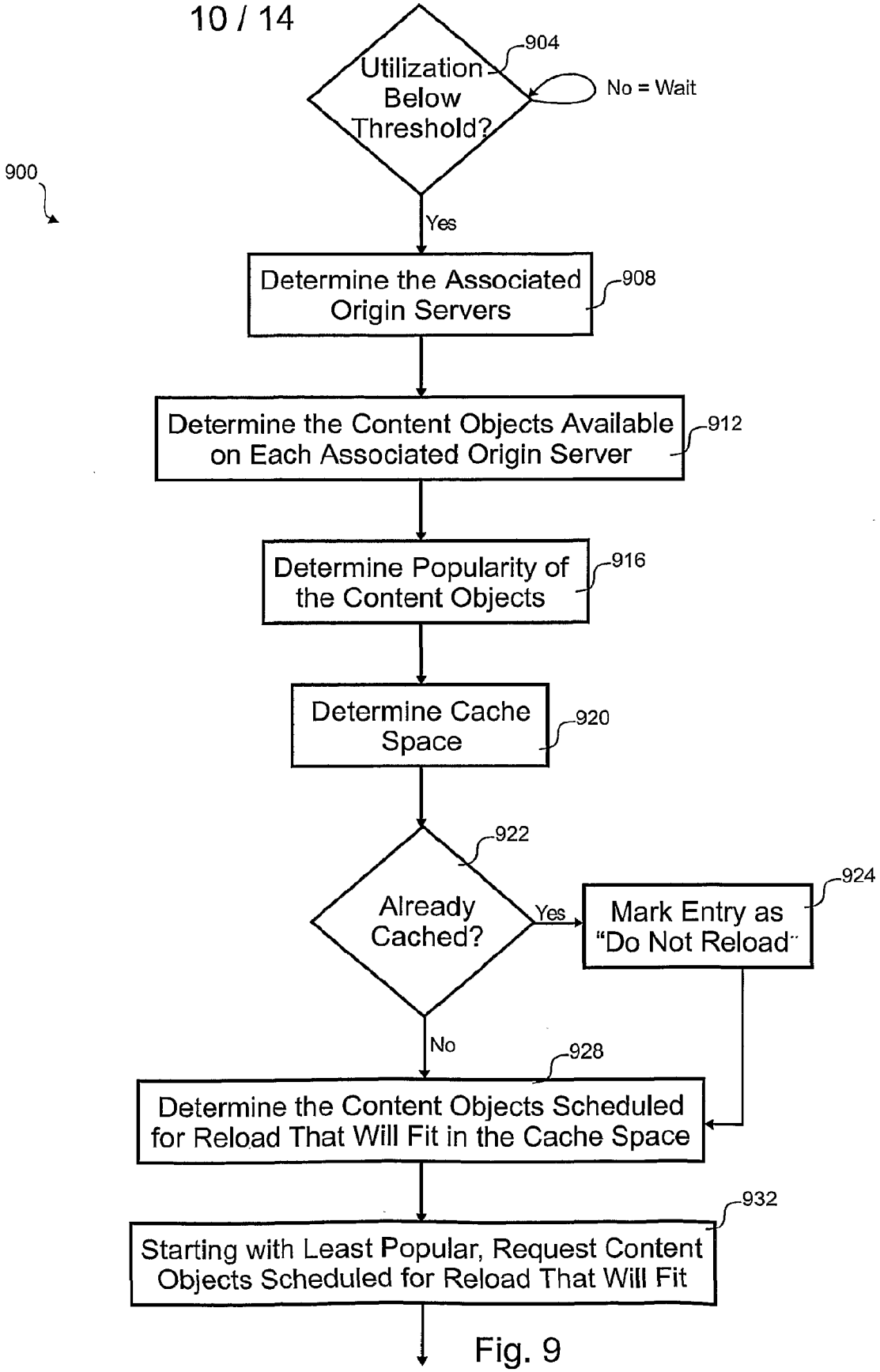

900
Utilization Below Threshold?
904
No = Wait
Yes
Determine the Associated Origin Servers
908
Determine the Content Objects Available on Each Associated Origin Server
912
Determine Popularity of the Content Objects
916
Determine Cache Space
920
Already Cached?
922
Yes
Mark Entry as "Do Not Reload"
924
No
Determine the Content Objects Scheduled for Reload That Will Fit in the Cache Space
928
Starting with Least Popular, Request Content Objects Scheduled for Reload That Will Fit
932

Fig. 9

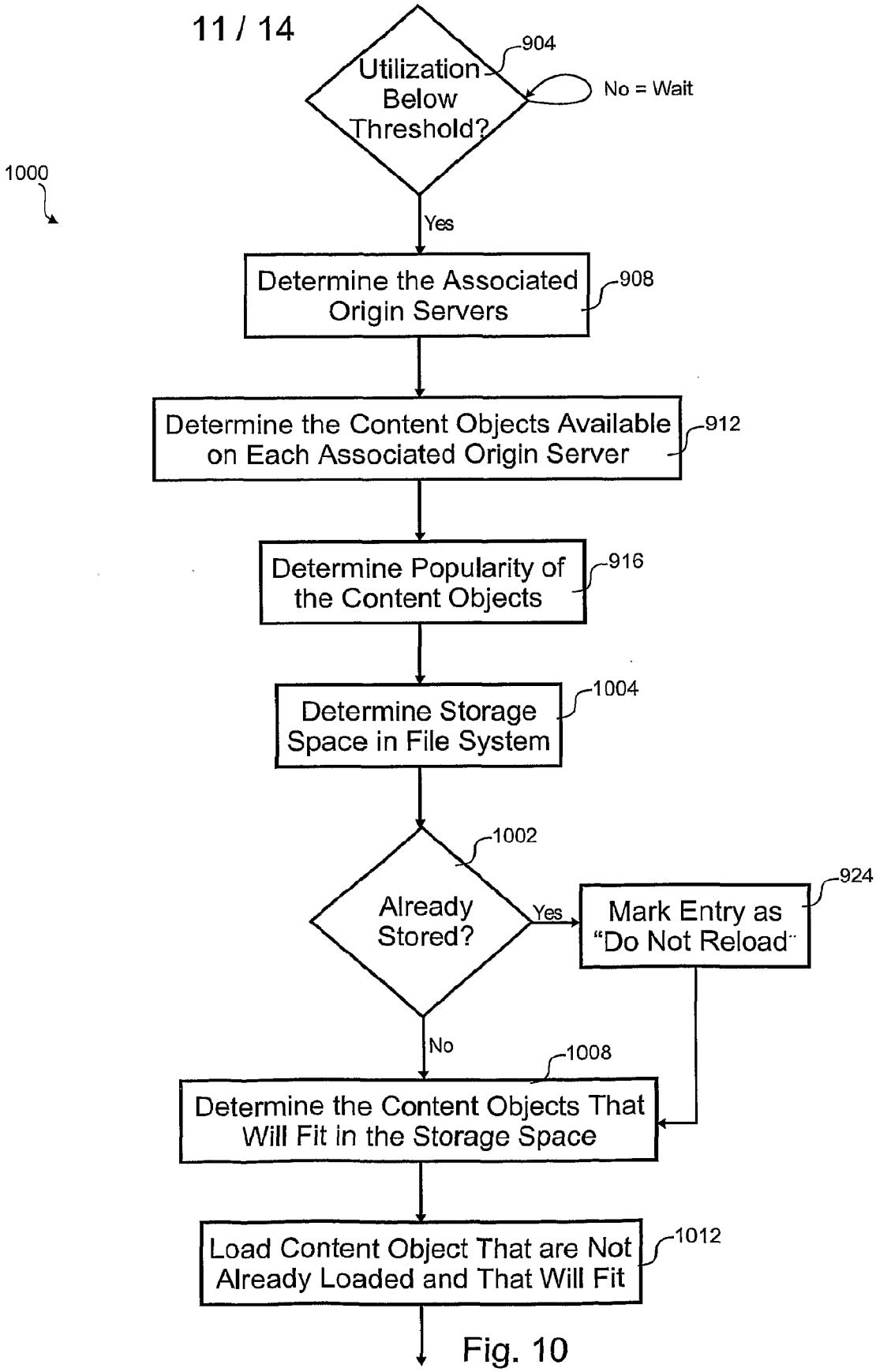

1000
Utilization Below Threshold?
904
No = Wait
Yes
Determine the Associated Origin Servers
908
Determine the Content Objects Available on Each Associated Origin Server
912
Determine Popularity of the Content Objects
916
Determine Storage Space in File System
1004
Already Stored?
1002
Yes
Mark Entry as "Do Not Reload"
924
No
Determine the Content Objects That Will Fit in the Storage Space
1008
Load Content Object That are Not Already Loaded and That Will Fit
1012

Fig. 10

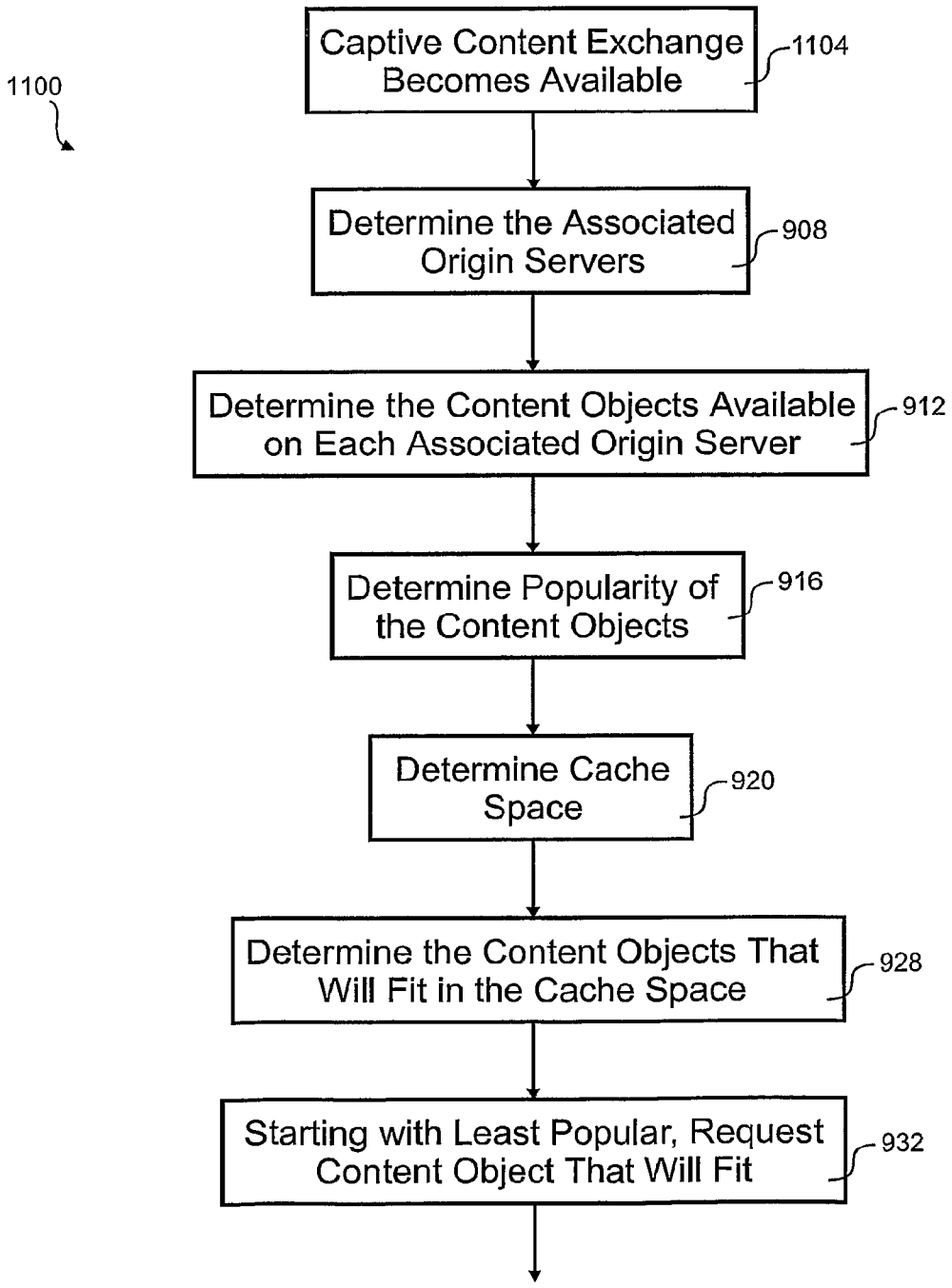

1100
Captive Content Exchange Becomes Available
1104
Determine the Associated Origin Servers
908
Determine the Content Objects Available on Each Associated Origin Server
912
Determine Popularity of the Content Objects
916
Determine Cache Space
920
Determine the Content Objects That Will Fit in the Cache Space
928
Starting with Least Popular, Request Content Object That Will Fit
932

Fig. 11

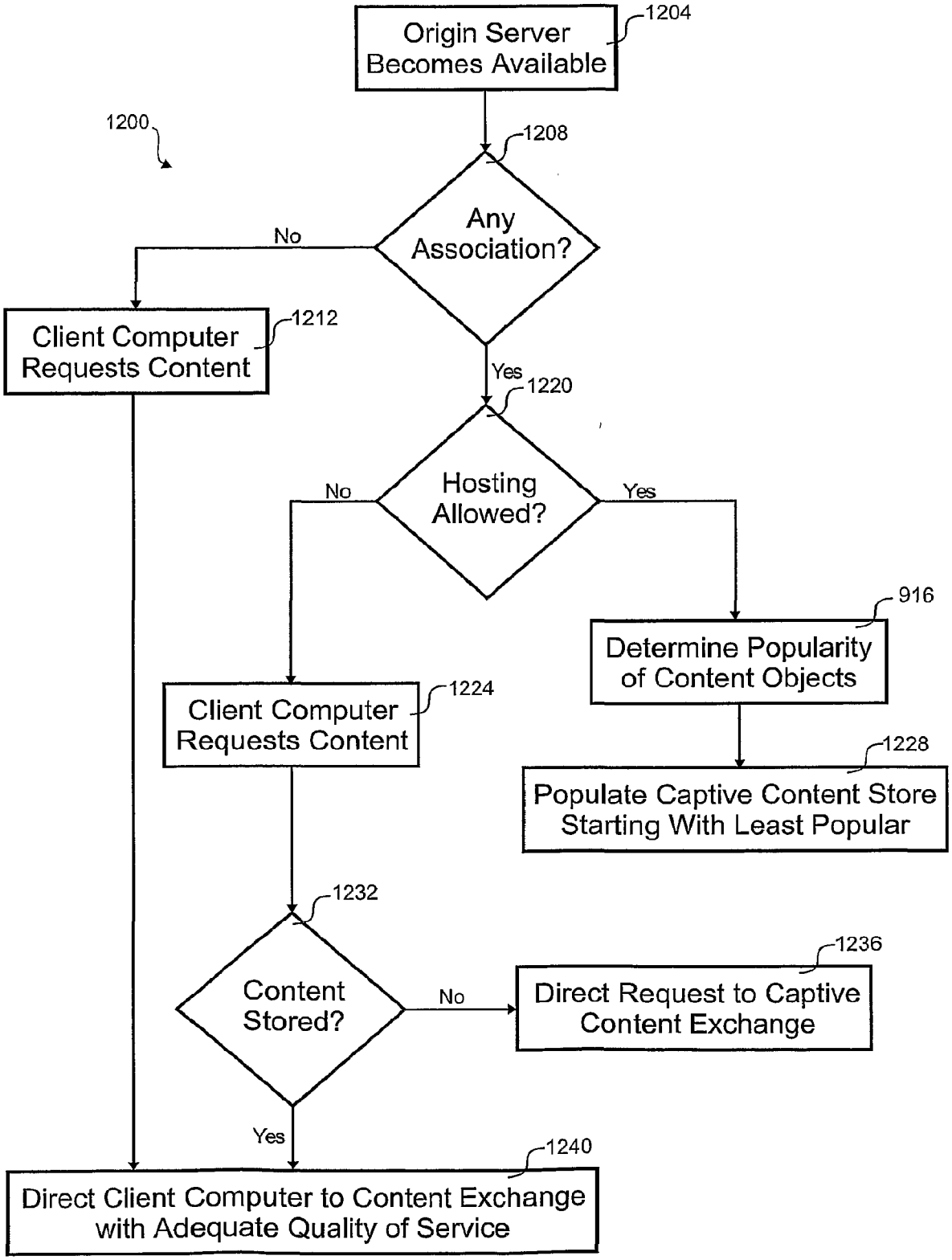

Origin Server Becomes Available
1204
1200
1208
Any Association?
No
Yes
Client Computer Requests Content
1212
1220
Hosting Allowed?
No
Yes
916
Determine Popularity of Content Objects
Client Computer Requests Content
1224
1228
Populate Captive Content Store Starting With Least Popular
1232
Content Stored?
No
1236
Direct Request to Captive Content Exchange
Yes
1240
Direct Client Computer to Content Exchange with Adequate Quality of Service

Fig. 12

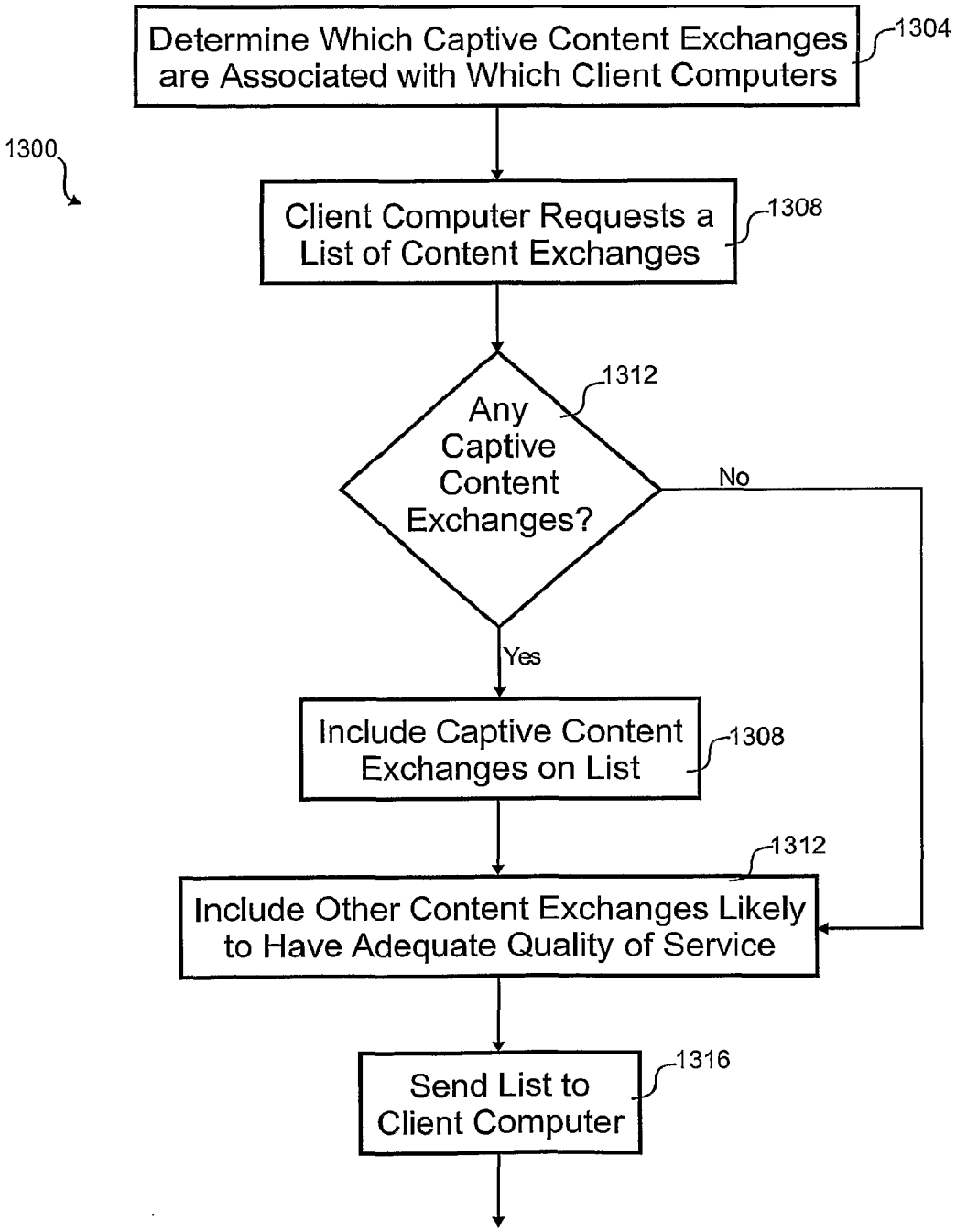

Determine Which Captive Content Exchanges are Associated with Which Client Computers
1304
1300
Client Computer Requests a List of Content Exchanges
1308
Any Captive Content Exchanges?
1312
No
Yes
Include Captive Content Exchanges on List
1308
Include Other Content Exchanges Likely to Have Adequate Quality of Service
1312
Send List to Client Computer
1316

Fig. 13

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2202572 C 2004/02/10

(11)(21) 2 202 572

(12) BREVET CANADIEN
CANADIAN PATENT

(13) C

(22) Date de dépôt/Filing Date: 1997/04/14

(41) Mise à la disp. pub./Open to Public Insp.: 1998/10/14

(45) Date de délivrance/Issue Date: 2004/02/10

(51) Cl.Int.^{6}/Int.Cl.^{6} G06F 15/16, G06F 13/38, H04L 29/02

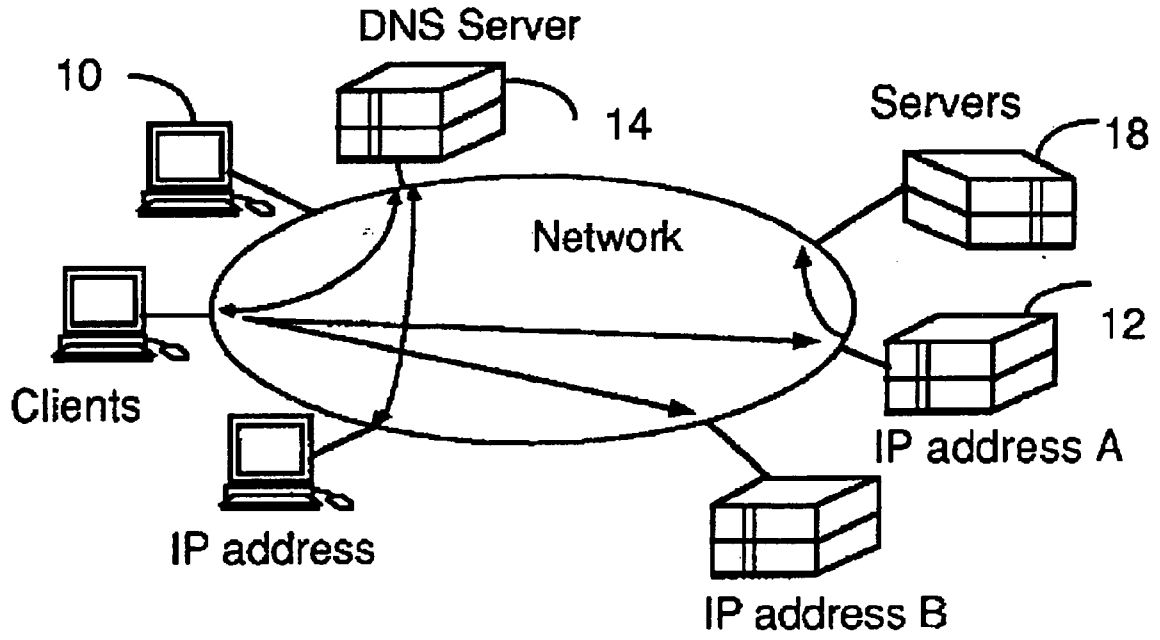
(72) Inventeurs/Inventors:
LAW, KA LUN EDDIE, CA;
NANDY, BISWAJIT, CA;
CHAPMAN, ALAN STANLEY JOHN, CA

(73) Propriétaire/Owner:
NORTEL NETWORKS LIMITED, CA

(74) Agent: GOWLING LAFLEUR HENDERSON LLP

(54) Titre : SERVEUR WEB A VARIABILITE DIMENSIONNELLE ET METHODE DE GESTION EFFICACE DE SERVEURS MULTIPLES

(54) Title: A SCALEABLE WEB SERVER AND METHOD OF EFFICIENTLY MANAGING MULTIPLE SERVERS



(57) Abrégé/Abstract:
A network includes one or more clients, a plurality of servers and an intermediate device called a "depot" Information resources are replicated among the servers The client accesses a server through the depot The depot transparently identifies a new session and selects one server for the session The depot may be provided between an Internet Client and an ATM network server. An edge node may be provided between the depot and the ATM server A proxy may be provided between the depot and a W W W server.

Abstract

A network includes one or more clients, a plurality of servers and an intermediate device called a "depot". Information resources are replicated among the servers. The client accesses a server through the depot. The depot transparently identifies a new session and selects one server for the session. The depot may be provided between an Internet Client and an ATM network server. An edge node may be provided between the depot and the ATM server. A proxy may be provided between the depot and a WWW server.

A SCALEABLE WEB SERVER AND METHOD OF EFFICIENTLY MANAGING MULTIPLE SERVERS

Field of the Invention

The invention relates generally to distributed server systems which are used widely in the field of telecommunications for information sharing. In particular, it is directed to a distributed server system which is scaleable and realizes high server performances. The distributed server system of the invention includes an intermediary node between a client and a pool of servers.

Background of the Invention

The last few years have observed a phenomenal growth in web (short for World-Wide Web or WWW, or Internet) usage. This growth has demonstrated the value of wide-area information-sharing but, at the same time, caused a significant research interest in improving the performance of web systems. Recent studies show that the web consumes more Internet bandwidth than any other application.

At a macro level, a web system consists of three components: (i) client, (ii) communication protocol, and (iii) server. Efforts are being made at each component to enhance the performance of the overall web systems.

At the client end, supports are provided to improve response time by the following features: memory cache, disk cache, allowing multiple simultaneous sessions and introducing a proxy server for another level of caching.

The communication protocol between the client and the server is HTTP (Hypertext Transfer Protocol) which always assumes the existence of a reliable path layer underneath the client and server. TCP/IP (Transmission Control Protocol/Internet Protocol) provides reliable data transmission using window flow control techniques. HTTP therefore runs on top of the TCP/IP layer. Asynchronous Transfer Mode (ATM) is another transmission technique to handle broadband multimedia traffic. It continues to grow steadily in the communication world. High-speed ATM switches are available in the commercial market. The co-existence of the Internet and large-scale ATM networks

is expected in the near future. ATM can provide wide-area virtual circuits, thus facilitating geographical distribution of web servers.

The HTTP has also undergone changes for performance improvement. It has been reported that multiple TCP sessions per HTTP transaction is a major cause of performance bottleneck. The introduction of a "keep alive" header allows sessions to be kept open and used for multiple HTTP request/response activities.

The web servers have also undergone improvements: first-generation servers handled 20 transactions/sec. based on one process per transaction. The major overhead was due to a large process fork time for a new transaction. This was avoided by pre-forking multiple processes and using a dispatcher to distribute transactions among them to achieve server performance of 100 transactions/sec. The "keep alive" HTTP feature, along with the multi-threaded architecture of only one process, allows the server to handle more than 250 transactions/sec. However, the current trend indicates that the popular sites will incur a significantly higher number of server transactions per second in the near future. This requires more powerful web servers which may be developed by improving different components of a web server (e.g., CPU speed, disk performance, file system performance, performance of TCP/IP, server software architecture etc.). Alternatively, multiple servers can be used to handle high rate of server transactions.

The multiple server approach has two immediate advantages: if a server fails, the session can be handled from other servers; also the total cost of multiple servers can be less than the cost of one server with the equivalent performance. It is therefore foreseen that multiple server systems will be in great demand to accommodate an ever-increasing number of user transactions.

Different architectures for multiple server systems are currently in use and are described briefly here. The use of a Domain Name System (DNS) server to distribute traffic among multiple servers was investigated at NCSA at the University of Illinois. Figure 1 shows the DNS system. When a client 10 wishes to communicate with a server 12, at first it contacts the DNS 14, from which it obtains the IP address of the desired server. The client then uses this IP address to communicate with the server. All clients perform the same process unless they

already have the IP addresses of servers with which they want to connect. When there are a plurality of servers which hold identical information, the DNS rotates in a round robin manner through a pool of these identical servers which are alternatively mapped to the alias of the hostname of one server. This approach has provided some success in distributing the server load, however, it could not balance the load among servers. Another problem with this approach is that, once the IP address resolution is cached in the local memory, the client may never contact DNS.

Another system uses the HTTP level redirection capability to move a transaction among multiple servers. Figure 1 also shows this mechanism. When a server 12 finds that it is impossible to handle any extra traffic, it can redirect a transaction to another preselected server 18 and hence distribute the load. HTTP redirection is a common technique used for WWW load distribution. The implementation maybe simple and straightforward, but the redirection requires a round trip delay between the client and server before the transaction is redirected to a different server. Moreover, if the first server is already very busy, the response delay will be even greater.

Figure 2 shows another known system which switches the load based on the client IP address. Each client 20 goes to an intermediate device 22 which examines the originating IP address and decides where to forward the traffic among multiple servers 24. IP address hashing is one of the possible mechanisms to determine the server to which the traffic will be directed. This technique, however, lacks the dynamic control of user accesses. Moreover, the IP address spaces are partitioned into five different classes. Care should therefore be taken in designing good hashing function.

HTTP is a stateless protocol. A web server obtains everything it needs to know about a request from the HTTP request itself. After the request is serviced, the server can forget the transaction. Thus, each request in HTTP is disjointed. If all the servers are identical (or see the same file system using a distributed file system), the server from which the request is served is of little relevance to the client. The choice of a physical server itself is immaterial to the transactions. An HTTP transaction is an aggregation of one or more TCP sessions. Based on this principle, different TCP sessions can be allocated to different

servers without the knowledge of whether or not all the TCP sessions belong to the same HTTP transaction. The present invention realizes this TCP-based switching by the use of an intermediary entity called a depot to perform these functions of session allocation. Thus, the TCP-based server switching allows a nice granularity for load balancing among multiple servers. It is also envisaged that this concept of forwarding different sessions to different servers can be applied to similar multi-server architectures of telecommunications networks.

Objects of the Invention

It is an object of the invention to provide a server system which has a high performance.

It is another object of the invention to provide a communications architecture which allows the construction of a scaleable and high performance server system.

It is a further object of the invention to provide a method of and an apparatus for efficiently managing the resources of a multiple server system.

It is still another object of the invention to provide a method of and an apparatus for efficiently managing the resources of a multiple server system by the use of an intermediate entity.

It is yet a further object of the invention to provide a method of and an apparatus for efficiently managing the resources of a multiple server system by the use of more than one instance of the intermediate entity for improved robustness.

Summary of the Invention

According to an aspect of the present invention, there is provided a method of efficiently utilizing the plurality of servers in a client-server environment of a telecommunications network, where information resources, which are accessed by HTTP protocol, are replicated among the servers and a client has access to any one or more of the servers for desired information resources. The method includes the steps of: performing an HTTP transaction by HTTP protocol between the client and the plurality of servers, the HTTP transaction comprising one or more TCP

connection sessions which are conducted by way of a depot; storing, at the depot, information about the TCP connection sessions and parameters in a primary map and a secondary map, the information in the primary map being deleted at the close of sessions and the information at the secondary map being maintained for two maximum segment lifetimes (MSL); transparently forwarding TCP connection sessions of the HTTP transaction at the depot in both directions to and from the client and the plurality of servers; monitoring parameters indicative of either or both of load and operability of the servers; detecting transition states of the TCP connection sessions of the HTTP transaction at the depot; and switching at the depot the plurality of TCP connection sessions of the HTTP transaction among the plurality of servers based on the detected transition states and the monitored parameters of the servers so that during the HTTP transaction, transfer of the information resources is performed by HTTP protocol between the client and one or more servers among the plurality of servers.

According to an aspect of the present invention, there is provided a method of efficiently utilizing the plurality of servers in a client-server environment of a telecommunications network, where information resources, which are accessed by HTTP protocol, are replicated among the servers and a plurality of clients have access to any one or more of the servers for desired information resources through a plurality of HTTP transactions. Each HTTP transaction uses multiple TCP connection sessions. The method includes the steps of: performing each HTTP transaction on HTTP protocol between any one of the plurality of clients and the plurality of servers, each HTTP transaction comprising one or more TCP connection sessions which are conducted on TCP/IP through a depot. The method further includes the following steps performed at the depot: storing information about the TCP connection sessions and parameters in a primary map and a secondary map, the information in the primary map being deleted at

the close of sessions and the information at the secondary map being maintained for two maximum segment lifetimes (MSL); inspecting all TCP/IP packets to detect a state transition of the TCP connection sessions; transparently forwarding TCP/IP packets of existing TCP connection sessions to a correct server; detecting TCP/IP packets of a new TCP connection session; monitoring parameters indicative of either one or both of load and operability of the servers; selecting a server among the plurality of servers based on their monitored parameters and the state transition; and transparently forwarding all the TCP/IP packets of the new TCP connection session to the selected server so that during each of the existing and new TCP connection sessions, transfer of the information resources of the HTTP transactions is performed between any one of the clients and one or more servers by one or more new and existing TCP connection sessions.

According to an aspect of the present invention, a depot for transferring one or more TCP connection sessions between a client and among a plurality of servers in a client-server environment of a telecommunications network, where information resources are replicated among the servers and a plurality of clients have access to any one or more of the servers for desired information resources through an HTTP transaction using at least two layers of protocols, one stateless protocol layer upon another stateful protocol layer. The HTTP transaction runs on the stateless protocol layer of HTTP protocol and is made up of one or more TCP connection sessions, which run on TCP/IP layer. The depot includes: a primary and a secondary map for storing information about the TCP connection sessions and parameters, the information at the primary map being deleted at the close of sessions and the information at the secondary map being maintained for two maximum segment lifetimes (MSL); a packet analysis block for analysing and transparently forwarding TCP/IP packets of one or more TCP connection sessions of the HTTP transaction between the

client and one of the servers; a session management block for switching one or more TCP connection sessions of the HTTP transaction among the plurality of servers according to a preset criteria including the TCP/IP states of each TCP connection session; and a protocol state tracking block for monitoring the TCP/IP states of the TCP connection sessions of the HTTP transaction.

Brief Description of the Drawings

Figure 1 shows multiple server systems of known architecture;

Figure 2 shows another multiple server system of known architecture;

Figure 3 shows schematically a client-server server system according to one embodiment of the invention;

Figure 4 is a functional block diagram of an intermediary device according to one embodiment of the invention;

Figure 5 is a schematic illustration of the invention as applied in the ATM environment according to one embodiment;

Figure 6 shows protocol stacks of the architecture shown in Figure 5;

Figure 7 is a functional block diagram of the embodiment shown in Figure 5;

Figure 8 is a schematic illustration of a yet another embodiment of the invention; and

Figure 9 shows schematically a yet further embodiment of the invention as applied in the proxy server environment.

Detailed Description of Preferred Embodiments of the Invention

Figure 3 shows schematically the invention according to one embodiment. A client 30 is communicating with a server 32 through a telecommunications network which can be any transport network such as telephone networks, data networks, ATM networks, LANs, or a combination thereof. WWW or Internet runs on top of the transport network. Typically, a client sends the server 32 a request for information stored therein and the server replies with the desired information. The desired information, however, may require more than one session (e.g., TCP session). In the multiple server environment, servers 34 and 36 contain identically stored information as that stored in server 32. According to the one embodiment of invention, an intermediate entity 38 sits transparently between the client and the servers. The intermediate entity 38 is called a "depot" throughout this specification and is the heart of the invention. When the depot receives a request for information, it distributes the TCP sessions among multiple servers based on the server load balancing criteria. In actual implementation, the depot can be realized by the use of appropriate software, hardware or a combination of the both. It should also be noted that the concept of session distribution among the servers can be implemented in other multiserver environments which use protocol stacks of similar configurations.

The depot is a forwarding mechanism and forwards any particular packet to the assigned server using a map. All clients access the server system using the depot's IP address. The distribution of TCP sessions among multiple servers by the depot remains transparent to the client. A TCP session consists of multiple TCP packets. All TCP packets of a given TCP session are served by the same server. Since the forwarding of all TCP packets of a given TCP session must go to the same server, a map between the IP address and port number of the

client and the identity of the server is maintained. The entry of this map is to be maintained in the depot as long as required by the TCP protocol. This entry is preserved in the depot as long as there is a possibility of the arrival of a TCP packet from a client or a server. The depot does not generate any TCP packets nor does it discard any unless they are anomalous enough that there is no way to assign them to a server.

A depot has the following functions:

1. inspect all packets in both directions at IP and TCP levels;
2. choose a server based on load balancing criteria for a new TCP session;
3. forward TCP packets for existing sessions to the already chosen server;
4. forward TCP packets from servers to clients;
5. clean up the mapping entry when TCP sessions end; and
6. watch for and handle anomalous TCP packets.

All these functions are performed in a depot and Figure 4 shows its functional block diagram. Referring to Figure 4, the depot includes TCP packet forwarding block 40 for TCP packet analysis and forwarding. This block inspects all packets in both directions at IP and TCP levels, forwards TCP packets of existing sessions to the already chosen server, and forwards TCP packets from servers to the clients. For a new session, this block identifies the TCP session setup request and forwards information to session management block 42. The session management block 42 chooses a server among a pool of available servers. This block is preloaded with the next server to be allocated. This pre-load is done by running a load balancing algorithm such as shown at 44 in the figure on the basis of knowledge of the server statistics and perhaps network states, or simply by round-robin. Server probing 48 can be performed periodically to obtain the server statistics. The TCP packets carrying data are forwarded by the TCP packet forwarding block 40 by identifying the entry of a map between the server and the client. At TCP state tracking block 46, TCP states of both the clients and the servers are being monitored to facilitate the management of the TCP session, e.g., opening/closing and control functions.

The depot maintains the map 50 of all the active TCP sessions in a table called the primary table. Each session between a client and a server is identified by the combination of the client's IP address and port number.

A typical primary table entry has the following: client IP address and port number, server identity, TCP states and related parameters (ack number, seq number etc.). The depot maintains another list of sessions in a table called the secondary table. The entries of the primary table are deleted at the close of sessions under normal or anomalous conditions. The secondary table entries are maintained for a 2 MSL (maximum segment lifetime) period. The duration of a 2 MSL differs for different implementations but, in this embodiment, a period of 2 minutes is chosen as an example. The 2 MSL is important since a TCP packet may arrive after a session is closed, due to variable delay at the network.

The reason for maintaining two separate tables at the depot is to reduce the search space for finding a match on arrival of a TCP packet. The number of entries in the secondary table is very large since the entries are maintained for a 2 MSL period. The entries in the primary table are for active TCP sessions only. Most of the incoming TCP packets should find a match in the primary table.

A TCP session is stateful within each session between a client and a server. Therefore, the depot should handle all arriving TCP/IP packets as transparently as possible; otherwise, the introduction of an intermediary such as the depot may disrupt the normal TCP logic between the two end systems. The crucial function of the state tracking routine in the depot involves the creation and deletion of the table entries. Several types of TCP/IP packets can initiate the creation and deletion of table entries. In the TCP state transition diagram, there are eleven states. For the client-server model, however, the number of possible states is reduced. This reduces the state tracking requirement in the depot. However, out-of-sequence situations can occur in the Internet, and different permutations of state transitions can arise which slightly increase the complexity of the state tracking routine. Moreover, in wide-area networks, packet loss is possible. Therefore, any packet received by the depot does not guarantee the reception of the packet by the other end. As a result, according to the invention,

packets in both directions are analysed in order to get a complete view of the state of the session and the state tracking function of the depot simply guesses the current TCP states of the client and server by investigating the "flag" information of all arriving TCP packets. The states guessed at the depot may be different from the actual states at the client and server, if a TCP packet is dropped or corrupted in the network between the depot and the client or server.

The following is a detailed description of the handling of various TCP packets according to one embodiment of the invention.

SYN: A SYN packet header (client IP address and port) is matched with the entries in the primary table and then with the secondary table. If no match is found (i.e., arrival of a new session), a new entry is created in the primary table and based on the load balancing criterion, a server is allocated for the session. If a match is found (i.e., duplicate SYN), the packet is forwarded to an already allocated server.

FIN, PSH, URG: The match is found from the primary or secondary table and forwarded to the appropriate server. If no match is found, the packet is dropped.

ACK: All ACK packets are forwarded to the server if an entry is found in the table. If the ACK packet causes the state transition to the TIME_WAIT state, the entry is moved from the primary table to the secondary table for a 2 MSL time-out.

RST: All reset packets are validated by checking if the sequence number is in the window. If the state is SYN-SENT, the RST is considered valid if the ACK field acknowledges the SYN. If the RST is valid, the entry from the primary table is moved to the secondary table.

The exception conditions are handled in the following manner.

If an entry in the primary table is inactive for a long time (e.g., more than 20 mins.), the entry is moved to the secondary table. This is necessary since a client or server may crash without proper termination of a TCP session, causing an entry to the primary table to remain forever.

If an entry in the tables (primary and secondary) is not found on arrival of a packet other than SYN, the packet is dropped. This is necessary since the depot does not know the destination for the packet.

The depot does not cause any interruption of the ongoing TCP sessions between the clients and the servers. This is because the depot forwards every packet, irrespective of the states guessed at the depot, as long as a table entry is found. For example, a RST may be lost in the network between the depot and client but the entry is still maintained in the secondary table, which enables the forwarding of any retransmitted RST. In one embodiment, the forwarding from the depot to the servers is achieved by directing packets to physical ports of the device containing the depot function where each physical port connects to one server.

According to a further embodiment, the depot system of the invention is implemented in the ATM environment. In this embodiment, an ATM network is used as a transport mechanism to provide distributed WWW services. The network model is shown in Figure 5. On the ATM network, multiple, information homogeneous, but potentially geographically distributed web servers 50 are provided. An Internet web client 52 can access information via the routes shown on the diagram. A depot 54 of the invention provides the interface between an Internet client and an ATM network server. An incoming packet from the client will be segmented and encapsulated into AAL5 cells at the depot without the requirement of modifying either the IP or TCP headers. The cells (packets) are carried sequentially in a virtual channel.

Like the earlier embodiment, the depot ideally sits transparently between the clients accessing from the Internet and the servers on the ATM networks. The depot employs a TCP-based switching mechanism. It examines all TCP/IP packets and sends them to the ATM network from the Internet and vice versa. Figure 6 shows the protocol stack structure of the system model. On the depot, there are TCP and IP stacks but the TCP session is only observed and not terminated.

Therefore, the depot of the embodiment performs the same functions as those described earlier. In addition, however, it must perform the following functions:

- transform IP packets to ATM cells and vice versa;
- map the QoS parameters from IP to ATM where applicable; and
- participate as necessary in the ATM network management.

Similar to Figure 4, Figure 7 shows a functional block diagram of the depot of this embodiment. The depot identifies the flow (client IP and port) from the TCP/IP packets and maps the corresponding VPI/VCI in its table entries. At the IP/ATM adaptation block, the depot performs segmentation and encapsulation of packets into AAL5 cells without the requirement of modifying either the IP or TCP headers. Likewise, it also perform the reverse functions of ATM cells from the ATM network. For a new session, the packet analyzer identifies the TCP session setup request and forwards the information to the session management block. The next server to be allocated is selected, as described earlier. Packets are kept contiguous and are all sent over the same VC.

If the session is already allocated and recognized by the packet analyzer, it will read the correct VPI/VCI from its tables and forward the packet onward. Packets in the reverse direction are also analyzed in order to have a complete view of the state of the session.

In another embodiment, the forwarding is achieved by encapsulating the IP frames in a frame transport, such as Ethernet or Frame relay, using the addressing of that frame transport to identify individual servers.

In another embodiment, the forwarding is achieved by modifying the header information of the TCP/IP packet and redirecting it over an IP network.

In Figure 8, another embodiment is shown. According to this embodiment, servers may or may not be ATM-based hosts and therefore an edge node 80 is required as an access point to each server, which provides traffic management and any necessary IP/ATM conversion. Figure 8 shows all the servers and Internet connections accessing the ATM network via edge nodes. The edge nodes communicate with each other over independent virtual circuits. Therefore, depot 82 is also an edge node managing the Internet connection that has the additional function of routing incoming packets to the correct server.

According to a yet further embodiment of the invention, the depot system of server management is applied to the proxy server architecture. Proxy servers are used to reduce the network load and latency by migrating the work load close to the clients.

A proxy server is an application gateway which operates at the HyperText Transfer Protocol (HTTP) layer. The basic function of a proxy server is almost identical to a HTTP server in transferring client requested documents. Furthermore, it is able to interpret and modify a HTTP request before forwarding it to the end WWW server. This is because the proxy server has the caching function. If there is a cache hit, it delivers the found documents to the client locally, thus reducing the network load and transmission latency.

Figure 9 shows schematically the concept of another embodiment of the invention as applied to the proxy server management system. In Figure 9, a depot proxy system is located between an intranet and the Internet. The depot distributes sessions among a pool of proxy servers based on load balancing or other criteria. The functions of the depot of the proxy system is identical to those described earlier. Therefore, for a new session, the packet analyzer identifies the TCP session setup request and forwards the information to the session management block. If the session is already allocated and recognized by the packet analyzer, it will read in the correct proxy identity from its tables and forward the packet onward. Packets in the reverse direction are also analyzed in order to have a complete view of the state of the session.

This proxy architecture can achieve the following similar goals:

1. scaleable proxy server arrangements;
2. high availability of information service; and
3. dynamic load balancing of traffic loads to different proxy servers.

To achieve scaleability, more proxy servers can be attached to the depot. This is because the depot handles only simple functions. When a depot operates to its limitations, it is also possible to add a new depot with a new IP address under the same alias name. With the newly added depot, another cluster of proxy servers can be created on the network.

Experimental Results

A Pentium PC running a NetBSD operating system is used for the software implementation of the depot. The client's HTTP requests are generated using benchmark software from Zeus Corporation. The clients and servers are connected to the depot using 10 Mbps Ethernet.

The same test file is retrieved 1000 times for each test. The number of simultaneous HTTP requests are varied to study the depot behaviour. Traffic is forwarded by the depot to two identical NCSA/1.5.1 Web servers in a round robin fashion. Server throughput (total bytes transferred per second) is measured for each test case. The total number of bytes of data and the HTTP headers divided by the time taken to transfer indicates the server throughput. The number of HTTP requests served by the server per second is also measured. Both measures include a variable network delay. To minimise the variable network delay, the experiment is performed at a time when the network is very lightly loaded. The variation of the server throughput and HTTP requests served per second with an increased rate of client requests are good indicators of server performance.

Table 1

	file size: 100 bytes		file size: 1 Kbytes		file size: 10 Kbytes	
Concurrent sessions	single server	invention	single server	invention	single server	invention
1	18.22	17.46	66.58	56.67	52.13	51.98
5	29.21	44.62	99.26	80.75	151.85	168.18
10	21.1	55.13	94.83	99.81	226.80	244.53
15	10.18	39.5	65.75	101.73	223.1	277.06
20	11.17	29.86	42.64	111.93	251.24	289.91
25	6.46	30.00	33.86	117.53	233.73	297.89
100	6.55	12.23	X	37.59	X	282.81

Table 2

	file size: 100 bytes		file size: 1 Kbytes		file size: 10 Kbytes	
Concurrent sessions	single server	invention	single server	invention	single server	invention
1	64.61	61.92	55.16	46.95	5.00	4.99
5	103.48	158.23	82.19	66.88	14.55	16.13
10	74.59	194.70	78.31	82.55	21.68	23.44
15	35.73	138.95	54.03	84.21	21.35	26.53
20	39.03	104.62	34.87	92.29	23.97	27.67
25	22.48	105.22	27.55	97.23	22.40	28.49
100	22.41	40.79	X	28.51	X	25.69

Table 1 shows the server throughput in Kbytes/sec. The experiment is performed with three different file sizes: 100 bytes, 1 Kbyte and 10 Kbytes. The number of simultaneous client HTTP requests are varied from 1 to 100. The total number of client HTTP requests for each test is 1000. The experiment is performed with a single server system and a depot system with two servers. Table 2 shows the number of served HTTP requests per second under the same experimental setup.

For one client HTTP request (any file size) at a time, the depot system is always slower than the single server. This is due to the store and forward delay introduced at the depot. For multiple simultaneous HTTP requests, however, the HTTP requests are served in parallel by two servers. In general, the depot system shows improved throughput and served HTTP requests per second. Another reason for performance improvement is the reduced workload at each server due to the distribution of HTTP requests. For 100 bytes and 1 Kbyte files, the single server performance degrades quickly with the increased number of simultaneous sessions. Performance with the depot system is consistently better than the single server system. For the 10 Kbytes file, the single server performance remains flat. However, the performance with the depot system is better than the single server system.

For 100 simultaneous HTTP requests, the single server system could complete the test for only a 100 bytes file size. The depot system could complete the tests for all three file sizes.

It is found that the throughput and HTTP requests served per second with the depot system are more than double those of the single server system with a large number of simultaneous sessions. For example, the number of HTTP requests served per second for the test case of 1 Kbyte file size with 20 simultaneous sessions is 92.29. This is an improvement of 2.65 over the single server system. This linear improvement is due to the halved number of simultaneous sessions (i.e., 10) on each server. The number of HTTP requests served per second by the single server system with 10 simultaneous HTTP requests is 78.31. Thus, maximum served requests of (2*78.31) 156.62 is theoretically attainable.

Thus it is shown that the server throughput and serviced requests per second can be improved using multiple servers according to the invention.

In the multiple server system, if one server goes down, the depot transfers all subsequent sessions to the other servers in the same cluster. There are several methods to detect server failure. A depot may assume the failure of a server if it does not receive an ACK from the server for several seconds after it sends a message to the server. After that, the depot starts polling the failed server on a regular basis to check if the server program has restarted. Another method that the depot uses is the passive reception of server statistics as an indication of the operability of a server program. If the depot does not receive any information for a period of time, then the depot will put this server into a dead-list and new incoming TCP requests will not be routed to this server. When server statistics are received from this server again, the depot will then put it back into an alive-list. However, there is difficulty in identifying the current state within each TCP session such as the number of bytes sent, the current window size and sequence number etc. Thus, the lost TCP sessions from a faulty server are simply scrapped.

The depot reliability is also very important since all the traffic is concentrated at the depot. Standby sparing is a simple method of providing fault tolerance in the system. There can be hot or warm standby. A sparing standby depot machine is always in alert mode to assume normal depot operations upon detecting a failure in the operating depot. For hot standby, the replica actively monitors all input streams from both the client and server. Both operating machine and replica have virtually identical TCP state information about all sessions. Detection of depot failure initiates changeover of control.

For warm standby, the operating machine periodically sends the state information to the replica to update the standby's database. In this situation, the two databases shall not be the same most of the time, but the goal is to reduce the number of affected TCP sessions when failure occurs. Then the standby replica will automatically be activated on fault detection.

What is claimed is:

1. In a client-server environment of a telecommunications network, where information resources, which are accessed by HTTP protocol, are replicated among a plurality of servers and a client has access to any one or more of the servers for desired information resources, a method of efficiently utilizing the plurality of servers comprising steps of:

performing an HTTP transaction by HTTP protocol between the client and the plurality of servers, the HTTP transaction comprising one or more TCP connection sessions which are conducted by way of a depot;

storing, at the depot, information about the TCP connection sessions and parameters in a primary map and a secondary map, the information in the primary map being deleted at the close of sessions and the information at the secondary map being maintained for two maximum segment lifetimes (MSL);

transparently forwarding TCP connection sessions of the HTTP transaction at the depot in both directions to and from the client and the plurality of servers;

monitoring parameters indicative of either or both of load and operability of the servers;

detecting transition states of the TCP connection sessions of the HTTP transaction at the depot; and

switching at the depot the plurality of TCP connection sessions of the HTTP transaction among the plurality of servers based on the detected transition states and the monitored parameters of the servers so that during the HTTP transaction, transfer of the information resources is performed by HTTP protocol between the client and one or more servers among the plurality of servers.

2. The method according to claim 1 wherein there are a plurality of depots, comprising further steps of:

monitoring parameters indicative of operability of all the depots; and

upon detection of inoperability of one depot, continuing all the functions of the failed depot at any of the remaining depots.

3. The method according to claim 1 wherein the depot is connected to one or more servers through an ATM network, comprising a further step of:

performing the TCP connection sessions in ATM cells through the ATM network.

4. The method according to claim 3 wherein there are a plurality of depots, comprising further steps of:

monitoring parameters indicative of operability of all the depots; and

upon detection of inoperability of one depot, continuing all the functions of the failed depot at any of the remaining depots.

5. The method according to claim 1 wherein the depot is connected to one or more servers through a local area network or a frame relay network, comprising further steps of:

performing the TCP connection sessions in their appropriate format through the local area network or the frame relay network.

6. The method according to claim 5 wherein there are a plurality of depots, comprising further steps of:

monitoring parameters indicative of operability of all the depots; and

upon detection of inoperability of one depot, continuing all the functions of the failed depot at any of the remaining depots.

7. The method according to claim 1, wherein the depot is connected to one or more servers by way of a plurality of proxy servers, comprising further steps of:

switching at the depot the TCP connection sessions among the proxy servers which in turn send the HTTP transaction to the servers; and

caching information resources contained in the HTTP transaction at the proxy servers.

8. The method according to claim 7, comprising further steps of:

monitoring the status of all the proxy servers; and

for each TCP connection session, selecting one proxy server among all the proxy servers based on their monitored status.

9. The method according to claim 8 wherein there are a plurality of depots, comprising further steps of:

monitoring parameters indicative of operability of all the depots; and

upon detection of inoperability of one depot, continuing all the functions of the failed depot at any of the remaining depots.

10. In a client-server environment of a telecommunications network, where information resources, which are accessed by HTTP protocol, are replicated among a plurality of servers and a plurality of clients have access to any one or more of the servers for desired information resources through a plurality of HTTP transactions, each HTTP transaction using multiple TCP connection sessions, a method of efficiently utilizing the plurality of servers during the HTTP transactions, comprising steps of:

performing each HTTP transaction on HTTP protocol between any one of the plurality of clients and the plurality of servers, each HTTP transaction comprising one or more TCP connection sessions which are conducted on TCP/IP through a depot; and

at the depot, performing the following steps:

storing information about the TCP connection sessions and parameters in a primary map and a secondary map, the

information in the primary map being deleted at the close of sessions and the information at the secondary map being maintained for two maximum segment lifetimes (MSL);

inspecting all TCP/IP packets to detect a state transition of the TCP connection sessions;

transparently forwarding TCP/IP packets of existing TCP connection sessions to a correct server;

detecting TCP/IP packets of a new TCP connection session;

monitoring parameters indicative of either one or both of load and operability of the servers;

selecting a server among the plurality of servers based on their monitored parameters and the state transition; and

transparently forwarding all the TCP/IP packets of the new TCP connection session to the selected server so that during each of the existing and new TCP connection sessions, transfer of the information resources of the HTTP transactions is performed between any one of the clients and one or more servers by one or more new and existing TCP connection sessions.

11. The method according to claim 10 wherein there are a plurality of depots, comprising further steps of:

monitoring parameters indicative of the operability of all the depots; and

upon detection of inoperability of one depot, continuing all the functions of the failed depot at any of the remaining depots.

12. The method according to claim 10 wherein the depot is connected to one or more servers through an ATM network, comprising a further step of:

performing the TCP connection sessions in ATM cells through the ATM network.

13. The method according to claim 12 wherein there are a plurality of depots, comprising further steps of:

monitoring parameters indicative of the operability of all the depots; and

upon detection of inoperability of one depot, continuing all the functions of the failed depot at any of the remaining depots.

14. The method according to claim 10 wherein the depot is connected to one or more servers through a local area network or a frame relay network, comprising further steps of:

performing the TCP connection sessions in their appropriate format through the local area network or the frame relay network.

15. The method according to claim 10, wherein the depot is connected to one or more servers by way of a plurality of proxy servers, comprising further steps of:

switching, at the depot, the TCP connection sessions among the proxy servers which in turn send TCP connection sessions to the server; and

caching information resources contained in the TCP connection sessions at the proxy servers.

16. The method according to claim 15, comprising further steps of:

monitoring the status of all the proxy servers; and

for each TCP connection session, selecting one proxy server among all the proxy servers based on their monitored status.

17. The method according to claim 16 wherein there are plurality of depots, comprising further steps of:

monitoring parameters indicative of the operability of all the depots; and

upon detection of inoperability of one depot, continuing all the functions of the failed depot at any of the remaining depots.

18. The method according to claim 14 wherein there are a plurality of depots, comprising further steps of:

monitoring parameters indicative of the operability of all the depots; and

upon detection of inoperability of one depot, continuing all the functions of the failed depot at any of the remaining depots.

19. In a client-server environment of a telecommunications network, where information resources are replicated among a plurality of servers and a plurality of clients have access to any one or more of the servers for desired information resources through an HTTP transaction using at least two layers of protocols, one stateless protocol layer upon another stateful protocol layer, the HTTP transaction running on the stateless protocol layer of HTTP protocol and being made up of one or more TCP connection sessions which run on TCP/IP layer, a depot for transferring one or more TCP connection sessions between a client and among a plurality of servers, comprising:

a primary map and a secondary map for storing information about the TCP connection sessions and parameters, the information at the primary map being deleted at the close of sessions and the information at the secondary map being maintained for two maximum segment lifetimes (MSL);

a packet analysis block for analysing and transparently forwarding TCP/IP packets of one or more TCP connection sessions of the HTTP transaction between the client and one of the servers;

a session management block for switching one or more TCP connection sessions of the HTTP transaction among the plurality of servers according to a preset criteria including the TCP/IP states of each TCP connection session; and

a protocol state tracking block for monitoring the TCP/IP states of the TCP connection sessions of the HTTP transaction.

20. The depot according to claim 19, further comprising:

a map which contains data concerning either one or both of load and operability of the plurality of servers; and

the session management block for switching one or more TCP connection sessions of the HTTP transaction among the servers based on the data stored in the map.

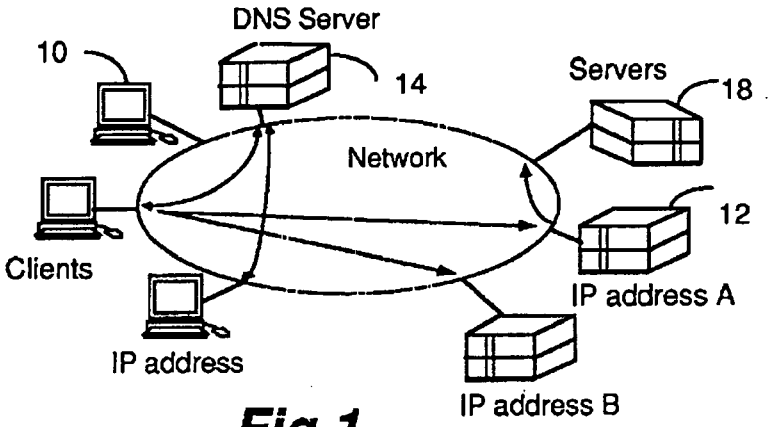


Fig 1

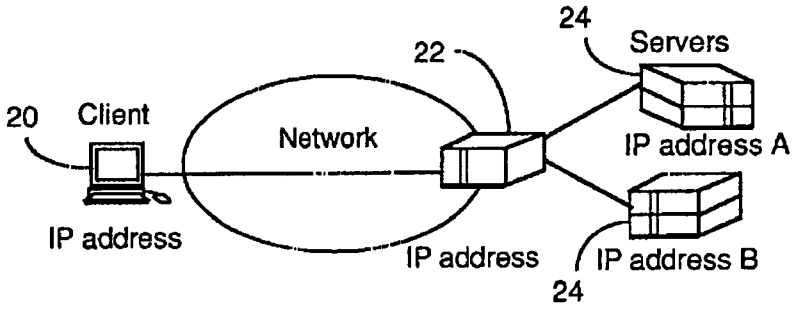


Fig 2

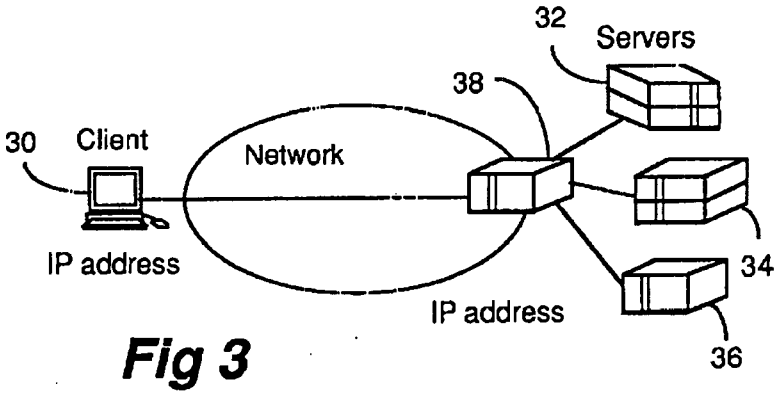


Fig 3

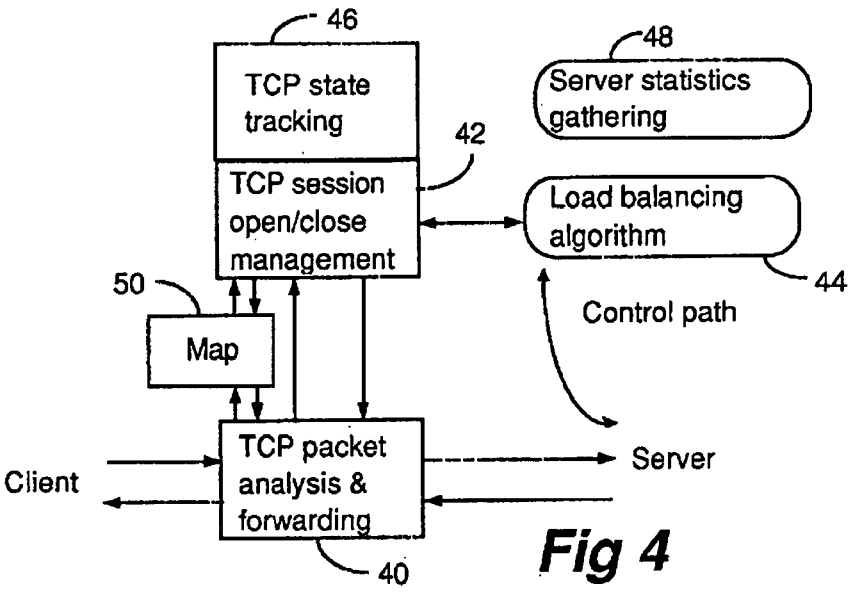

46
TCP state tracking
48
Server statistics gathering
42
TCP session open/close management
Load balancing algorithm
44
50
Map
Control path
Client
TCP packet analysis & forwarding
Server
40

Fig 4

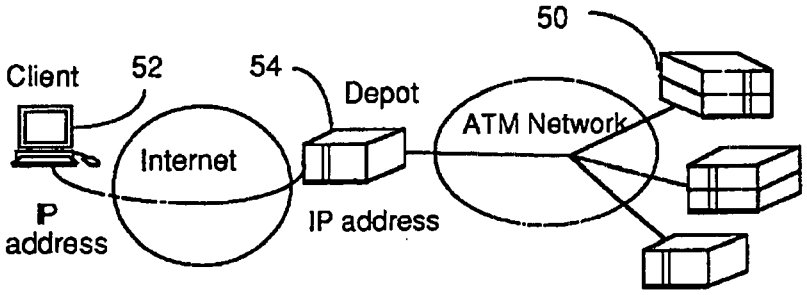

50
Client
52
54
Depot
ATM Network
Internet
IP address
IP address
Pool of servers for IP address A

Fig 5

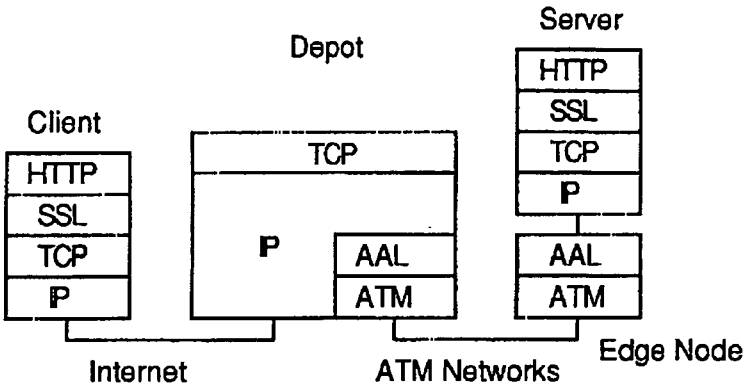


Fig 6

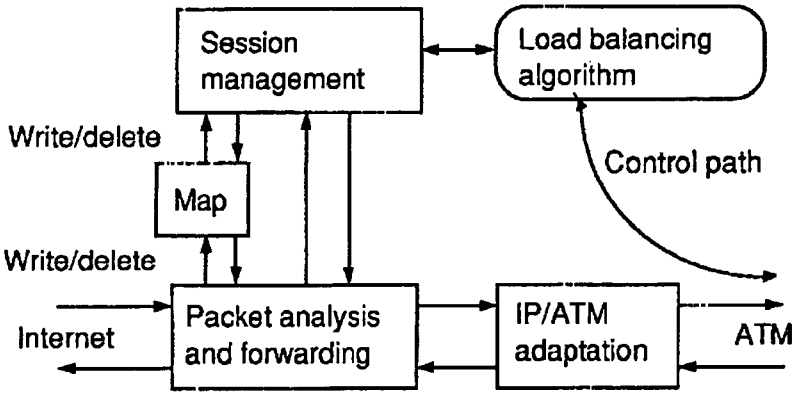


Fig 7

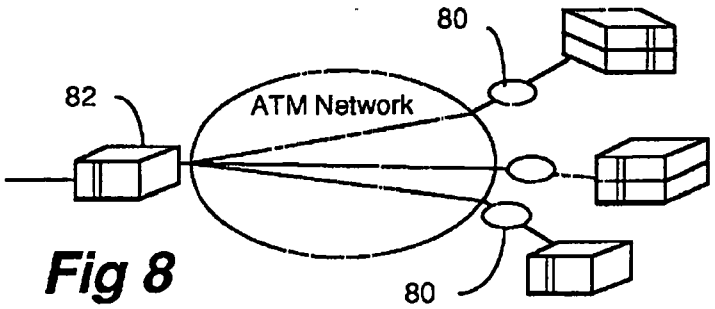


Fig 8

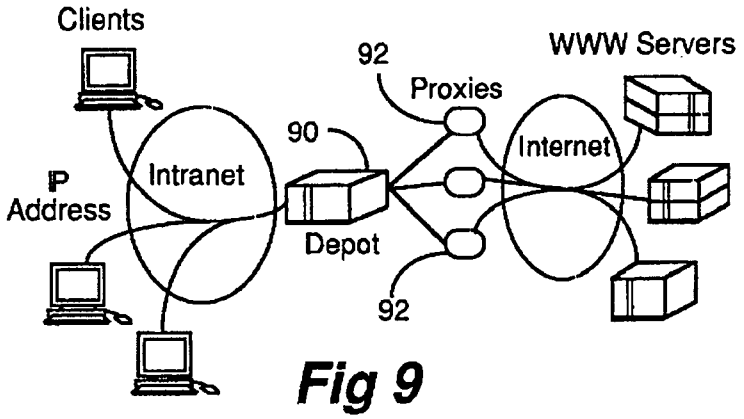

Clients
92
Proxies
WWW Servers
90
Internet
IP
Address
Intranet
Depot
92

Fig 9

(12) UK Patent Application (19) GB (11) 2 281 793 (13) A

(43) Date of A Publication 15.03.1995

(21) Application No 9318877.9

(22) Date of Filing 11.09.1993

(71) Applicant(s)
International Business Machines Corporation

(Incorporated in USA - New York)

Armonk, New York 10504, United States of America

(72) Inventor(s)
Graham Derek Wallis
Michael George Taylor
Michael Platt
Andrew James Stanford-Clark

(51) INT CL[6]
G06F 15/16

(52) UK CL (Edition N)
G4A AFN

(56) Documents Cited
EP 0240145 A2 EP 0232509 A2

(58) Field of Search
UK CL (Edition L) G4A AFN
INT CL[5] G06F 15/16
ONLINE DATABASES : WPI

(74) Agent and/or Address for Service
G A Bailey
IBM United Kingdom Limited, Intellectual Property Department, Mail Point 110, Hursley Park, WINCHESTER, Hampshire, SO21 2JN, United Kingdom

(54) A data processing system for providing user load levelling in a network

(57) A data processing system 10 facilitates connection of a program on a client computer 20 to a server, the server consisting of a plurality of server computers 40, 50, 60 with shared resources. The system 10 has an input means 30 for receiving a request from the client computer for a machine address of a named server computer, for enabling a connection to be made from the client computer to that server computer. A device 80 stores a list identifying server computer names with associated machine addresses. A conversion means 70 uses the list to convert the server computer name received by the input means 30 into the machine address, which an output means 110 sends to the client computer. The system is characterised by: decision logic 120 for studying the server computers at predetermined intervals having regard to predetermined test criteria, e.g. their loading, in order to select one of the server computers; and writing means 160 for updating the list by associating the machine address for the server computer selected by the decision logic with a particular server computer name contained as a generic server computer name in the list. Using this technique, when a client computer specifies the generic server computer name, it receives the machine address of the server computer identified by the decision logic.

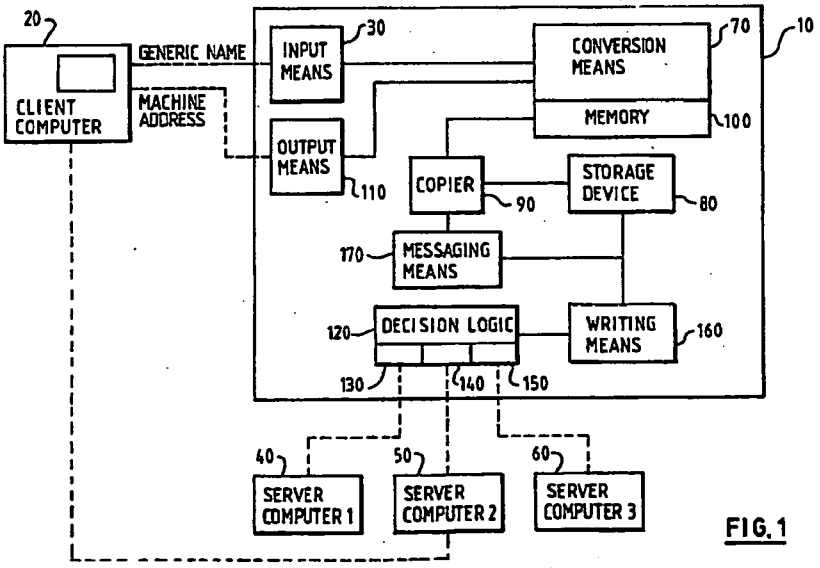


FIG.1

GB 2 281 793 A

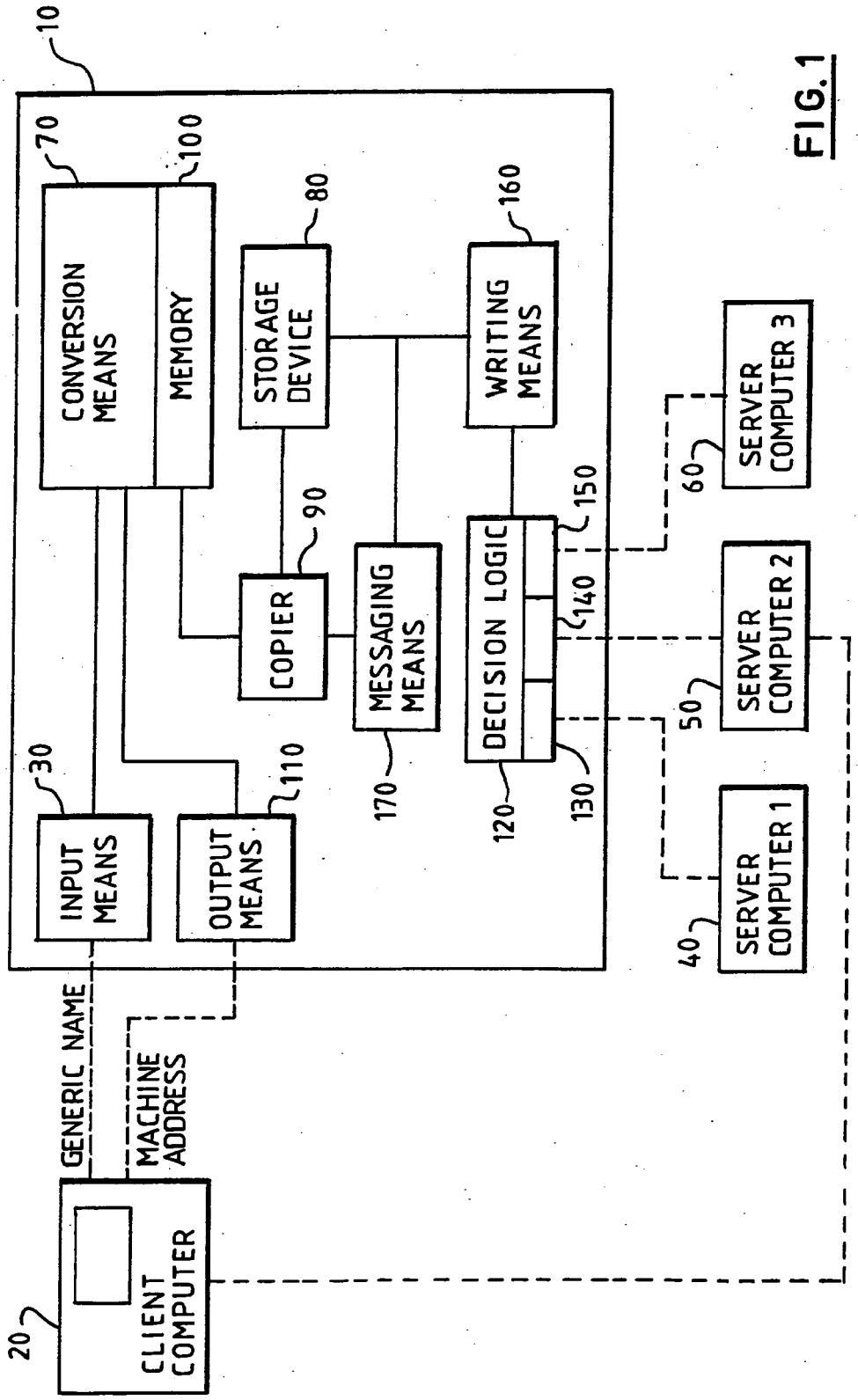

10
70
100
CONVERSION MEANS
MEMORY
80
STORAGE DEVICE
160
WRITING MEANS
SERVER COMPUTER 3
60
150
90
COPIER
MESSAGING MEANS
DECISION LOGIC
140
SERVER COMPUTER 2
50
30
110
170
120
130
INPUT MEANS
OUTPUT MEANS
SERVER COMPUTER 1
40
GENERIC NAME
MACHINE ADDRESS
CLIENT COMPUTER
20

FIG. 1

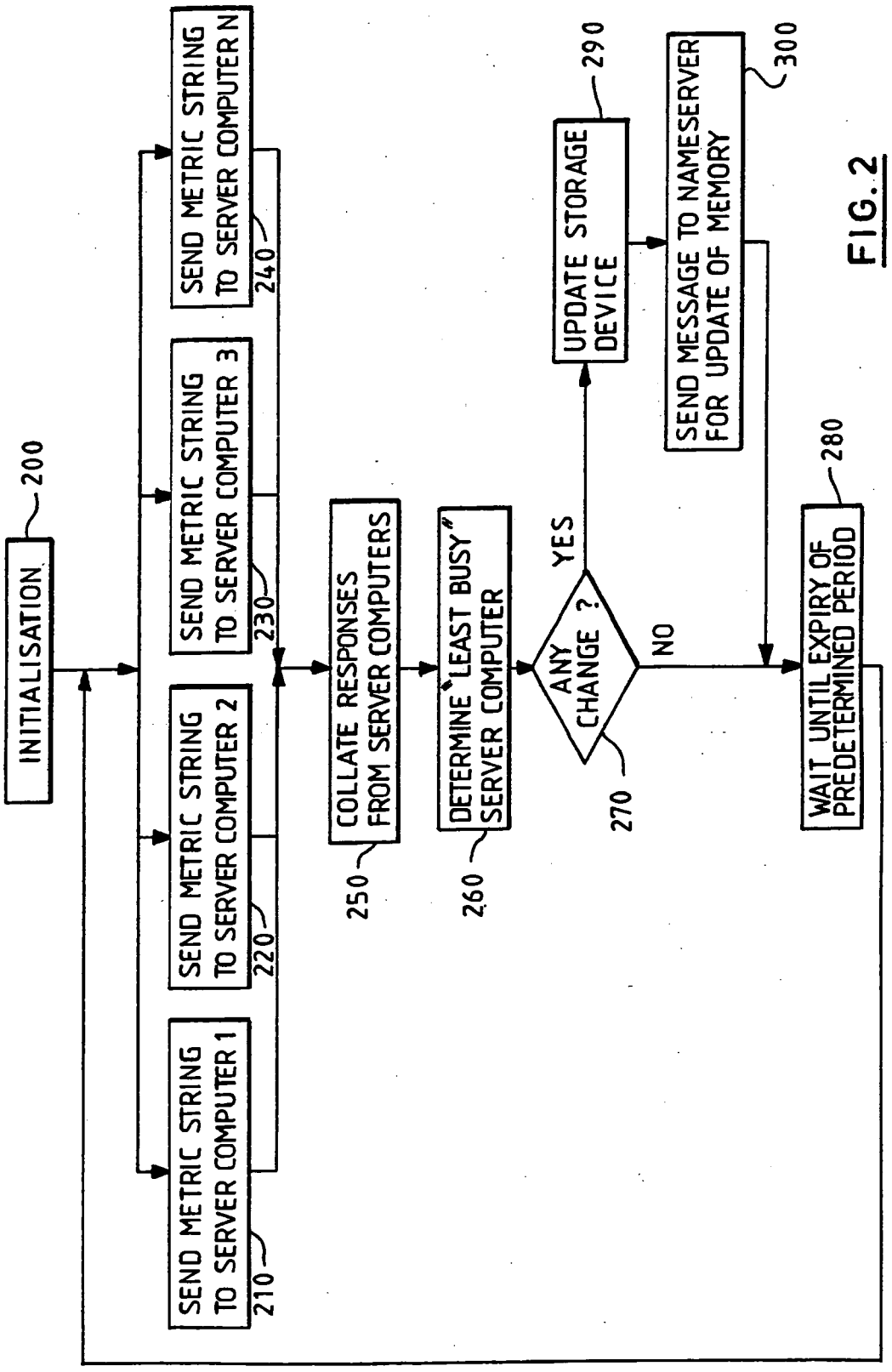

INITIALISATION
200
SEND METRIC STRING TO SERVER COMPUTER 1
210
SEND METRIC STRING TO SERVER COMPUTER 2
220
SEND METRIC STRING TO SERVER COMPUTER 3
230
SEND METRIC STRING TO SERVER COMPUTER N
240
COLLATE RESPONSES FROM SERVER COMPUTERS
250
DETERMINE "LEAST BUSY" SERVER COMPUTER
260
ANY CHANGE ?
270
YES
NO
UPDATE STORAGE DEVICE
290
SEND MESSAGE TO NAMESERVER FOR UPDATE OF MEMORY
300
WAIT UNTIL EXPIRY OF PREDETERMINED PERIOD
280

FIG. 2

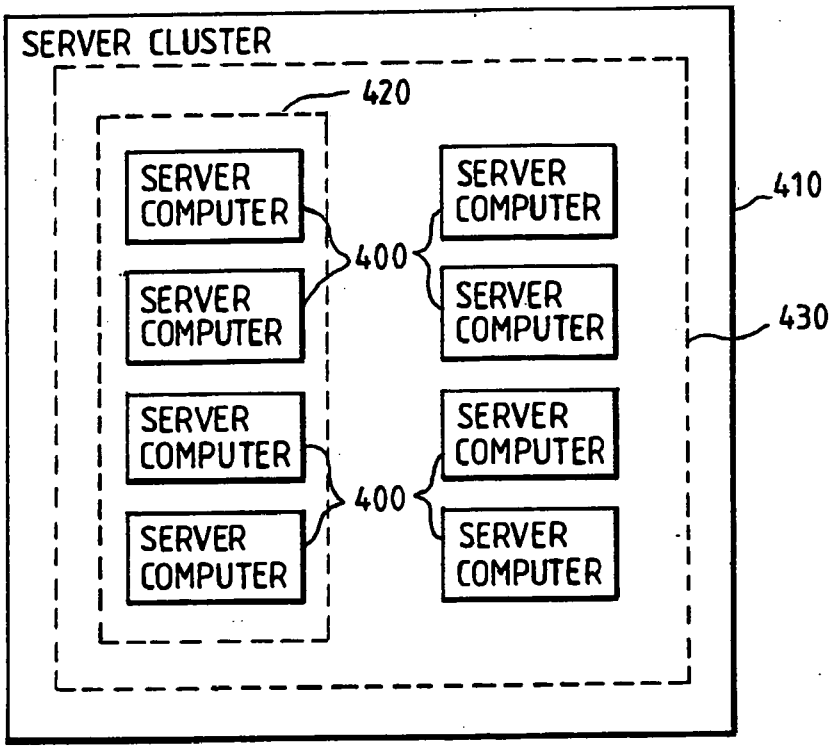

SERVER CLUSTER
420
SERVER COMPUTER
SERVER COMPUTER
SERVER COMPUTER
SERVER COMPUTER
400
SERVER COMPUTER
SERVER COMPUTER
SERVER COMPUTER
SERVER COMPUTER
400
410
430

FIG. 3

A DATA PROCESSING SYSTEM FOR PROVIDING USER LOAD LEVELLING IN A NETWORK

The present invention relates to a data processing system for facilitating the connection of a program on a client computer to a server, the server consisting of a plurality of server computers with shared resources.

The data processing system, the client computer, and the server computers are all resident on a network. This network need not be one physical network such as a Local Area Network (LAN); for instance it may consist of a number of LANS or WANS (wide area networks) connected together (eg. via 'bridges') to form a single logical network. However the same network protocol will be employed throughout the network, a typical example of such a protocol being TCP/IP which will be familiar to those skilled in the art.

In many environments it is increasingly found that a number of server computers are connected together using some form of network, eg. a LAN. It is often the case that several users of client computers will be connected to one such server computer, whilst other server computers stand idle. An example of this is when such computers are situated in workers' offices, connected by, for example, a token ring LAN using the standard TCP/IP network protocol. When workers are away from their offices, their computers will usually stand idle.

In such situations it is commonly the case that a few of the computers in the network are heavily loaded, whilst other computers in the network are very lightly loaded, giving poor response and performance for the client computers using the heavily loaded server computers. Hence there is a problem of how to enable the client user load to be spread more evenly across the available computing resources of the server in a manner which is transparent to the client computer and its programs. If transparency is to be achieved, standard protocols need to be observed in order that client computers can use a variety of connection methods without modification of any programs being required.

A prior art technique which has been developed to provide some sort of load spreading is called "Static load levelling". With this technique

each application on each client computer has a designated server to which it always connects. Hence, for example, if there are 200 potential clients of a server having five server computers, a pre-specified group of, say, 40 of the clients will be told (or configured) to always connect to machine 1, etc. On average it may be argued that this will give a reasonably even load across all of the server computers. However in practice it is often the case that, using this technique, a large number of users of client computers are connected to one server computer, while an adjacent server computer stands completely idle (eg. given the above example there could easily be 40 users on one server computer whilst the other 4 server computers stand idle). Hence in situations where the user loading changes from time to time, the prior art static load levelling technique is not particularly satisfactory. What is required in such instances is a more 'dynamic' technique which can respond to changing user loads, and thus direct new users to the most suitable (eg. least heavily loaded) server computers in the server.

Other prior art techniques can be found in other environments, such as those where job allocation is an issue. For instance in "batch processing", a client computer submits an encapsulated task to a central server, which determines which one of several possible servers is quiet enough to be able to handle the task. The task is sent to that server, is processed, and the results are then sent back to the client (e.g. as a results file, or by electronic mail). With a batch processing system, there is a brief connection to the central server while the job is transferred from the client to the server allocated by the central server. After this brief connection the client disconnects, and has no more interaction with the submitted task until it has been completed, and the results have been passed back to the client by some means.

However in the situation with which we are currently concerned, the dynamic load levelling technique that is required must be able to deal with "interactive" sessions. Rather like a phone conversation, the connection between the program on the client computer and a particular server computer will persist for the duration of the "conversation" session. Hence the batch processing concept is inappropriate in the present situation.

It is possible to write some specific code within a program on a client computer which contains internal message-passing systems to route

work from that client program to a corresponding server program. Such systems are dedicated only to that particular client program, and the connection and load-levelling methods are not accessible to other client-server applications. Often, such systems operate by the client connecting to a specific "host" server computer, and from there the work will be sent to another server for processing. Clearly this technique can result in large bottlenecks arising at the "host" server computer.

Hence such a technique is not suitable in the present situation since it only supports one very specific type of client-server connection, whereas we need a technique that will allow any client-server connection method using the network protocol to be connected to a quiet server in a way that is completely transparent to the client program. Further the above technique relies on an initial connection to the 'central' host server computer, which then passes the request on to another server computer; as described above this can potentially create a serious bottleneck.

It is an object of the present invention to provide a technique which facilitates a connection between a client program and a server computer on a server in a way that takes into account the current status of the server computers forming the server. This technique must observe standard network protocols and should operate in a manner which is transparent to the client program requesting access.

Accordingly the present invention provides a data processing system for facilitating a connection of a program on a client computer to a server, the server consisting of a plurality of server computers with shared resources, the data processing system, the client computer, and the server computers residing in a network, the system comprising: input means for receiving a request from the client computer for a machine address of a server computer identified by a server computer name sent with the request, such a machine address enabling a connection to be made from the client computer to that server computer via the network; a storage device for storing a list identifying server computer names with machine addresses of the server computers; conversion means for using the list to convert the server computer name received by the input means into the machine address of the server computer; output means for sending the machine address from the conversion means to the client computer; the system being characterised by: decision logic for studying the server

computers at predetermined intervals having regard to predetermined test criteria, in order to select one of the server computers; and writing means for updating the list by associating the machine address for the server computer selected by the decision logic with a particular server computer name contained as a generic server computer name in the list; whereby when a client computer specifies the generic server computer name, it receives the machine address of the server computer identified by the decision logic.

Typically the conversion means will access the list from a local piece of storage, the data processing system having a copier to copy the list from the storage device to that piece of memory. In preferred embodiments the data processing system further comprises a messaging means, responsive to the updating of the list by the writing means, for sending a message to the copier requesting the copier to copy the updated list into the piece of local memory.

Any manner of predetermined test criteria can be used in the data processing system of the invention, for example the amount of idle processor time, the number of processes running, the amount of free memory, the "load average", etc. However in preferred embodiments the predetermined test criteria are such that the decision logic identifies the server computer having the least number of client programs logged on to it.

In preferred embodiments the predetermined intervals are variable and will either be set by a user of the system, eg. the system administrator, or will be adjusted dynamically. The user will also set the predetermined test criteria to be used by the decision logic.

Further in preferred embodiments the user can limit the number of server computers which the decision logic studies. This may be useful if, for instance, some of the server computers have not got access to all of the resources that other server computers have access to, and so would not be suitable as server computers to be associated with the generic server computer name.

In some embodiments it may be advantageous to use a plurality of generic names. Each server name would then have a number of server computers whose machine addresses are associated with that generic name,

the decision logic employing different sets of predetermined test criteria for each generic name. In such embodiments one or more of the server computers can be associated with a plurality of the generic names.

Viewed from a second aspect the present invention provides a method of operating a data processing system to facilitate a connection of a program on a client computer to a server, the server consisting of a plurality of server computers with shared resources, the data processing system, the client computer, and the server computers residing in a network, the method comprising the steps of: (a) receiving a request from the client computer for a machine address of a server computer identified by a server computer name sent with the request, such a machine address enabling a connection to be made from the client computer to that server computer via the network; (b) storing a list identifying server computer names with machine addresses of the server computers in a storage device; (c) converting, with reference to the list, the server computer name received at step (a) into the machine address of the server computer; (d) sending the machine address identified at step (c) to the client computer; the method being characterised by the steps of: (e) employing decision logic to study the server computers at predetermined intervals having regard to predetermined test criteria, in order to select one of the server computers; and (f) updating the list by associating the machine address for the server computer selected by the decision logic with a particular server computer name contained as a generic server computer name in the list; whereby when a client computer specifies the generic server computer name at step (a), it receives the machine address of the server computer identified by the decision logic.

The present invention will be described further, by way of example only, with reference to an embodiment therof as illustrated in the accompanying drawings, in which:

Figure 1 is a block diagram illustrating the data processing system of the preferred embodiment;

Figure 2 is a flow diagram illustrating how the decision logic in the data processing system of the preferred embodiment operates; and

Figure 3 illustrates a particular embodiment where two generic computer names are used.

In the preferred embodiment we will consider the situation where the server in question is a high performance database server which has its data distributed across a network of server computers, this server network hereafter being referred to as a cluster. Database applications being run by users on client computers are required to connect to one of the server computers in the cluster to enable them to access the data in the database server. By the nature of the database system, it does not matter which server computer the client connects to - all of the data is accessible from any server computer in the cluster. In the preferred embodiment the server computers and the client computers are all interconnected using TCP/IP on a token ring Local Area Network.

For a large number of users, it is highly desirable to have a number of users on each of the server computers in the server cluster, rather than all users connecting to (and hence overloading) just one or a few of the server computers. With a widely varying user workload profile for the database server, this problem can only be solved by providing some form of "load levelling" process, which will allocate new client application instances to server computers in the cluster that are most suitable for the client connection (eg. because they are more lightly loaded than other server computers). Clearly this process must be dynamic, able to respond to changing load conditions over time. Since the database applications on the client computers are typically complex and often are supplied only in object code form, it would be very difficult (or impossible) for the system administrator to alter them, and so it is essential that this allocation is done in a manner which is entirely transparent to the client application.

The manner in which the data processing system of the preferred embodiment solves the above problems will now be described with reference to Figure 1.

Each client computer in a network using the TCP/IP protocol (there will typically be many such client computers) will have been informed by the network administrator that it is to communicate with a particular computer when it wishes to convert a computer name of another computer in the network into a machine address. When utilising the present invention that computer will be the data processing system of the preferred embodiment.

Hence when a program running on a client computer 20 (for clarity, only one client computer is illustrated) wishes to obtain access to a server computer (40, 50, 60) in the cluster it will communicate with the data processing system 10 in order to obtain a full Internet machine address for the desired server (Internet addressing is part of the TCP/IP protocol). With the prior art technique the client computer would specify a server computer name in this communication that was specific to one particular server computer in the cluster. The input means 30 of the data processing system 10 would receive this server computer name and pass it to the conversion means 70.

In a storage device 80 of the data processing system a list is maintained which identifies server computer names with particular Internet addresses. When the conversion means is initiated the copier 90 copies this list from the storage device 80 into a piece of local memory 100 accessible by the conversion means 70. Hence the conversion means will access the list in memory 100 to find the Internet address of the computer associated with the server computer name passed to it by the input means 30. This Internet address will then be provided by the conversion means to the output means 110 for transmission back to the client computer 20.

Once the client computer has the Internet address it can then make direct contact with the server computer residing at the Internet address provided; in Figure 1 this is server computer 2. Since the TCP/IP protocol is used any of the access methods that use this protocol can be used to access the server computer.

When using the data processing system of the preferred embodiment the program running on the client computer 20 will not use the server computer name that it previously used. Instead a generic server computer name will be used. This generic server name will either have been placed in the program's configuration file, or alternatively the user of the program will specify the generic name when running the program.

Within the data processing system, decision logic 120 is provided which periodically studies the server computers in the cluster having regard to some predetermined test criteria, hereafter called the metric string. In the preferred embodiment the metric string is a list of questions which when answered by the various server computers will enable

the decision logic to decide which server computer is most suitable for a client connection (the most suitable perhaps being the least heavily loaded server computer). The metric string can be altered as the system administrator deems appropriate, depending on what criteria the administrator wishes to be used to select a server computer.

In the preferred embodiment the decision logic actually sets up a number of child processes, each one being responsible for sending the metric string to a particular server computer and receiving the response from the server computer.

Once the responses have been received the decision logic will collate the responses, decide which server computer is most suitable, and then request the writing means 160 to pass the Internet address of that server computer to the storage device for association with the generic server computer name. If however the most suitable server computer is the same server computer as that identified in the last iteration of the process then there is no need to update the storage device and the writing means will not be activated.

Once any necessary update to the storage device 80 has been made the messaging means 170 will notify the copier 90 so that the copier updates the local memory 100 with the new list as stored in the storage device 80.

Hence when the client computer requests a machine address for the generic server computer name the conversion means 70 accesses the list in memory 100 and identifies a machine address just as it would if any other server computer name had been given. However in this instance the machine address actually relates to the server computer in the cluster which has been identified by the decision logic as the most suitable (eg. least heavily loaded). When this machine address is passed back to the client computer 20 via the output means 110, the client computer will proceed to automatically access the server which is most suitable.

By this approach it will be seen that a dynamic load levelling facility is provided which is completely transparent to the client program. As far as the program is concerned it is requesting a machine address as normal and is using one of the normal TCP/IP access methods to gain access to the server computer allocated to it.

In many of todays computing environments (eg Unix, AIX (Unix is a Trade Mark of Unix Systems Laboratories Inc)) an application is provided to perform the standard name resolution service (ie receipt of a computer name and conversion of that computer name into a full Internet address). This application is commonly known as a "nameserver" application, and is installed on one or more computers in the logical network. Every other computer in the network is told to communicate with a specified one of these 'nameserver' computers when it wishes to determine an Internet address for any other computer in the network. Hence a nameserver computer provides a resolution service to client computers by receiving from them a convenient name given to a particular computer (eg. abc.def.ghi.com), and converting it into a full Internet address (eg. 29.1.19.66). This Internet address is then used by the routing subsystem (TCP/IP) to allow a client user or application access to the physical computer (eg "abc" in this example).

In the above example of a computer name, "abc" is the physical machine, "def" is typically the site location, "ghi" the organisation, and "com" one of the Internet classes (three such classes are (com)mercial, (edu)cation, (mil)itary). Domains and sub-domains can also be added as part of this computer name. Basically the name takes a hierarchical form, with the finest resolution at the beginning and the coarsest resolution at the end; this type of naming structure will of course be well known to those skilled in the art.

All TCP/IP-based applications, including remote-login, remote-shell, telnet, ftp, and also client-server applications (such as database applications), are aware of the nameserver facility, and will automatically go to the designated nameserver computer to ask for resolution of a computer name into an Internet address before attempting to make a connection to another computer in the network.

If we consider Figure 1 again, the standard nameserver facility will include the following elements: the input means 30, the conversion means 70 with associated memory 100, the output means 110, the list stored in the storage device 80, and the copier 90.

The nameserver application is a "daemon" (background) process which runs on the data processing system; this data processing system may (but need not) be one of the server computers forming part of the cluster over

which users are to be distributed. In Unix-type operating systems (eg. AIX by IBM Corporation, Ultrix by Digital Equipment Corporation, OSF/1 by the Open Software Foundation, and HP-UX by Hewlett Packard, etc) this daemon process is called "named" (name-daemon), and when it is initialised, it reads a special database file (named.data) stored on the storage device 80 to obtain details of the computer names about which it is expected to know (over which it has "authority"), and the corresponding Internet addresses ("dotted decimal", e.g. 29.1.19.66) for each computer name. Whilst the name daemon is operating, it can be forced to re-read the information from the named.data database file by the sending of an inter-process signal to the name daemon process telling it to update its internal tables 100 from the database file (named.data).

In the preferred embodiment of the present invention we provide a further facility which runs on the same computer as the nameserver application ("named"), and interfaces with it. A 'generic' computer name is introduced into the database file (named.data), which refers not to one specific computer, but to any one of a number of computers offering equivalent functionality. For example, the generic name might be "server.cluster.def.ghi.com"; a client program requesting a connection to 'server.cluster' is requesting connection to any one of the computers in the server cluster.

The further facility provided by the preferred embodiment will be referred to hereafter as the "User Load Leveller" (ULL) application. This application is responsible for deciding which server computer in the cluster is currently the least heavily loaded, according to some appropriate metric, and for conveying this information to the nameserver application. Then subsequent requests for resolution of the generic server computer name to an Internet address result in the nameserver application sending back to the client computer the Internet address of the server computer which has been deemed to be the most appropriate server computer for connection at that point in time.

The ULL application consists of the following elements from figure 1: the decision logic 120 with child processes 130, 140, 150; the writing means 160; and the messaging means 170. As described earlier with reference to Figure 1 the ULL application periodically (at a frequency which can varied (eg. tuned by a system administrator or dynamically adjusted)) polls the server computers in the cluster to determine how

"busy" in some sense they are. The metric used may vary, depending on the type of work which is being handled by the cluster, but may for example include the number of login sessions, number of application instances running, number of idle cpu cycles since the last poll, etc. The metric can be altered to ensure that it is appropriate to a specific situation.

Based on the results of this polling, and taking into account the situation where a server computer in the cluster is too busy to respond to the status request within a certain number of seconds, the ULL application decides which machine is currently the least heavily loaded. The ULL application then modifies the database file (named.data) to associate the generic cluster machine name with the Internet address of this least heavily loaded machine, and sends the special inter-process signal via the messaging means 170 which tells the nameserver application to re-read its database file. The nameserver application will then, in response to a name resolution request from a client program, resolve the generic server computer name into the Internet address of the most appropriate server computer in the cluster for the client program to connect to.

The process carried out by the decision logic 120 of the preferred embodiment will now be described in more detail with reference to Figure 2. At step 200 the ULL application is initialised. A number of steps are carried out at initialisation. For example the application: checks for multiple copies of the ULL application in memory; cleans up from a previous run of the application (by freeing up system resources such as memory, locks and semaphores still held in the name of the previous instance of the ULL application); and locates the nameserver application (named) and its data file (named.data). The ULL application then parses its configuration file to read information defined by the system administrator, such as the metric strings, poll periods, identities of server computers in the cluster, etc. Further the ULL application generates a number of "child" processes - one per server computer - which are each responsible for polling the activity of one designated server.

Once the initialisation has been completed the process enters a main loop which executes until the ULL application is terminated. At steps 210, 220, 230 and 240 the child processes send a metric string (as defined by the system administrator) to each server computer in the

cluster, await responses from those computers, and then wait for a trigger signal from the main ULL application.

Once the trigger signal has been sent by the main application the responses are sent by the child processes to the main application. The main application then collates the activity results received from the child processes (step 250), and based on predetermined test criteria identifies the most appropriate server computer (the "least busy" server computer) at step 260. At step 270 it is determined whether the server computer identified at step 260 differs from the current nominated server computer. If it does then the process advances to step 290, at which point the nameserver's data file (named.data) is modified. Further at step 300 a notification signal is sent to the nameserver application (named) to tell it to update its internal information from the data file.

The process then proceeds to step 280. If at step 270 it is determined that the server computer identified at step 260 is the same as the current nominated server computer then the process moves straight to step 280 without steps 290 and 300 being carried out. Writing to and reading from the data file are time consuming activities and so steps 290 and 300 should only be performed when necessary (ie when the "least busy" server computer changes).

At step 280 the process waits until the end of the "poll period". This period is the predetermined interval (as defined by the system administrator) between successive studies of the server computers by the ULL application. Once the poll period has expired the process loops back to steps 210-240 and the main loop is repeated.

Having discussed the preferred embodiment a few possible alterations will now be discussed. Firstly more than one generic server computer name can be added to the list in storage device 80 (the named.data file). Each generic name could be associated with a particular group of server computers, these groups being either completely separate or having a few server computers common to a plurality of the groups. Indeed one group may be a subset of another group. As an example consider Figure 3. A server cluster 410 comprises eight server computers 400. All eight server computers have access to a main body of data, but only four of them have access to some further (possibly more confidential) data.

In this situation two generic names could be generated, eg. "general.cluster" and "specific.cluster". Any one of the eight computers (enclosed by ring 430) can be associated with the former generic name, but only the four enclosed by ring 420 can be associated with the latter generic name, since only those four have access to the further (confidential) data.

The system administrator can then set up the metric string to be used when studying all eight server computers, or when studying only the four in ring 420; the metric string could be the same in both instances but need not be. If a client application needs access to the confidential information then it would request access to "specific.cluster", but if an application only needed access to the general information then it would request access to "general.cluster".

By this approach an application which only needs access to the general information will always be connected to the least busy server computer, whilst an application which needs access to the further (confidential) information will be given the machine address of the least busy server computer that can actually provide the necessary service; this may or may not be the least busy server computer in the network.

In preferred embodiments a further feature is provided to enable the decision logic to temporarily implement a "round-robin" metric instead of the above described 'studying' process. The round-robin principle will be familiar to those skilled in the art; basically when a client application requests access to a server computer it is assigned a particular server computer, and when the next request is received then that application is assigned the next server computer in the cluster, and so on. In this way the server computers are rotated so that each successive server access is made on a different server computer to the previous server access. Alternatively the server computers can be rotated at fixed time intervals rather than after each access request.

Although the round-robin technique does not have regard to the loading on any of the server computers, and so there is no determination of the least busy server computer, there are certain situations (eg. where there are lots of client applications which only take a short amount of database connection time) where a round-robin approach is acceptable. To implement the round robin approach the decision logic 120

would ask the writing means 160 to update the storage device 80 after each access request has been handled (or at fixed time intervals if the alternative approach is used), so that the generic name is always associated with successive server computers in the cluster in turn.

From the above description it will be clear that the system of the preferred embodiment has a number of advantages. Firstly the technique dynamically allocates new client users and applications to the server computer which is least heavily loaded at the time they make the connection, thus ensuring an even distribution of users and applications across all of the available server computers. The client computer only briefly contacts the data processing system of the preferred embodiment to resolve the generic computer name into a machine address. Completely standard access methods (eg. as provided by TCP/IP) are then used to make the connection, thus avoiding any proprietary protocols or any need to modify access methods or applications, and so providing fully transparent user load levelling.

Secondly the technique of the preferred embodiment does not involve any modification to the nameserver code - the User Load Leveller application interfaces with the standard code (eg. "named" as shipped with the unix/AIX operating system). It would be possible to provide similar functionality to that described here by producing a modified version of the nameserver code. However, avoiding this brings major advantages from both marketing and maintenance points of view.

Further the technique can be operated without requiring any modification to the server computers. They are accessed in a standard way after the generic server computer name has been used to provide the client computer with a machine address.

Another advantage is that the key parameters, such as the time interval between polls of the server computers in the cluster and the metric used to determine which server computer is least heavily loaded, can be altered and tuned by a local system administrator, allowing the system to be optimised for a particular situation.

The above described ULL application could be supplied as a separate tool to enhance the useability of parallel and distributed systems, or could be shipped with the nameserver application.

CLAIMS

1. A data processing system for facilitating a connection of a program on a client computer to a server, the server consisting of a plurality of server computers with shared resources, the data processing system, the client computer, and the server computers residing in a network, the system comprising:

input means for receiving a request from the client computer for a machine address of a server computer identified by a server computer name sent with the request, such a machine address enabling a connection to be made from the client computer to that server computer via the network;

a storage device for storing a list identifying server computer names with machine addresses of the server computers;

conversion means for using the list to convert the server computer name received by the input means into the machine address of the server computer;

output means for sending the machine address from the conversion means to the client computer;

the system being characterised by:

decision logic for studying the server computers at predetermined intervals having regard to predetermined test criteria, in order to select one of the server computers; and

writing means for updating the list by associating the machine address for the server computer selected by the decision logic with a particular server computer name contained as a generic server computer name in the list;

whereby when a client computer specifies the generic server computer name, it receives the machine address of the server computer identified by the decision logic.

2. A system as claimed in Claim 1 further comprising:

a copier to copy the list from the storage device to a piece of memory accessible by the conversion means;

a messaging means, responsive to the updating of the list by the writing means, for sending a message to the copier requesting the copier to copy the updated list into the piece of local memory.

3. A system as claimed in Claim 1 or Claim 2, wherein the predetermined test criteria are such that the decision logic identifies the server computer having the least number of client programs logged on to it.

4. A system as claimed in any of claims 1 to 3, wherein the predetermined intervals are variable.

5. A system as claimed in any preceding claim, wherein the predetermined test criteria are set by a user of the system.

6. A system as claimed in any preceding claim, wherein the user can limit the number of server computers which the decision logic studies.

7. A system as claimed in any preceding claim wherein a plurality of generic names are used, each one having a number of server computers whose machine addresses are associated with that generic name, the decision logic employing different sets of predetermined test criteria for each generic name.

8. A system as claimed in Claim 7, wherein one or more of the server computers are associated with a plurality of the generic names.

9. A method of operating a data processing system to facilitate a connection of a program on a client computer to a server, the server consisting of a plurality of server computers with shared resources, the data processing system, the client computer, and the server computers residing in a network, the method comprising the steps of:

(a) receiving a request from the client computer for a machine address of a server computer identified by a server computer name sent with the request, such a machine address enabling a connection to be made from the client computer to that server computer via the network;

(b) storing a list identifying server computer names with machine addresses of the server computers in a storage device;

(c) converting, with reference to the list, the server computer name received at step (a) into the machine address of the server computer;

(d) sending the machine address identified at step (c) to the client computer;

the method being characterised by the steps of:

(e) employing decision logic to study the server computers at predetermined intervals having regard to predetermined test criteria, in order to select one of the server computers; and

(f) updating the list by associating the machine address for the server computer selected by the decision logic with a particular server computer name contained as a generic server computer name in the list;

whereby when a client computer specifies the generic server computer name at step (a), it receives the machine address of the server computer identified by the decision logic.

10. A method as claimed in Claim 9 further comprising the steps of:

copying the list from the storage device to a piece of memory accessible at the conversion step (c);

repeating, in response to the updating of the list at step (f), the copying step to ensure that the updated list is copied into the piece of local memory.

11. A method as as claimed in claim 9 or claim 10, wherein the predetermined intervals are set by a user of the system.

12. A method as claimed in any of claims 9 to 11, wherein the predetermined test criteria are set by a user of the system.

13. A method as claimed in any of claims 9 to 12, wherein the user can limit the number of server computers which the decision logic studies.

14. A method as claimed in any of claims 9 to 13, wherein a plurality of generic names are used, each one having a number of server computers whose machine addresses are associated with that generic name, the decision logic employing different sets of predetermined test criteria for each generic name.

15. A method as claimed in Claim 14, wherein one or more of the server computers are associated with a plurality of the generic names.

Patents Act 1977 Examiner's report to the Comptroller under Section 17 (The Search report) -19-	Application number GB 9318877.9
Relevant Technical Fields (i) UK Cl (Ed.L) G4A (AFN) (ii) Int Cl (Ed.5) G06f 15/16	Search Examiner B G WESTERN Date of completion of Search 18 October 1993
Databases (see below) (i) UK Patent Office collections of GB, EP, WO and US patent specifications. (ii) On-line database : WPI	Documents considered relevant following a search in respect of Claims :- 1-15

Categories of documents

X: Document indicating lack of novelty or of inventive step.

Y: Document indicating lack of inventive step if combined with one or more other documents of the same category.

A: Document indicating technological background and/or state of the art.

P: Document published on or after the declared priority date but before the filing date of the present application.

E: Patent document published on or after, but with priority date earlier than, the filing date of the present application.

&: Member of the same patent family; corresponding document.

Category	Identity of document and relevant passages		Relevant to claim(s)
A	EP 0232509 A2	IBM (see whole document)	
A	EP 0240145 A2	TOSHIBA (see whole document)	

Databases: The UK Patent Office database comprises classified collections of GB, EP, WO and US patent specifications as outlined periodically in the Official Journal (Patents). The on-line databases considered for search are also listed periodically in the Official Journal (Patents).

(19) Canadian Intellectual Property Office
An Agency of Industry Canada

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

(11) CA 2 288 488 (13) A1

(43) 16.06.2000

(12)

(21) 2 288 488

(51) Int. Cl.[7]: H04L 029/06, H04L 012/56

(22) 03.11.1999

(30) 09/212,980 US 16.12.1998

(71) LUCENT TECHNOLOGIES INC.,
600 Mountain Avenue, MURRAY HILL, XX (US).

(72) SINGH, NAVJOT (US).
COHEN, ARIEL (US).
RANGARAJAN, SAMPATH (US).

(74) KIRBY EADES GALE BAKER

(54) METHODE ET DISPOSITIF D'ACHEMINEMENT TRANSPARENT DES DEMANDES D'OBJETS WEB VERS UN CACHE DE MANDATAIRE

(54) METHOD AND APPARATUS FOR TRANSPARENTLY DIRECTING REQUESTS FOR WEB OBJECTS TO PROXY CACHES

(57)

In order to transparently redirect an HTTP connection request that is directed to an origin server (107) to a proxy cache (110-1), a proxy redirector (104) translates the destination address of packets directed to the origin server to the address of the proxy. During a handshaking procedure, a TCP connection is transparently established between the client (110-1) and the proxy cache. When the client transmits a GET request to what it thinks is the origin server, which request specifies the complete address of an object at that origin server that it wants a copy of, the proxy redirector modifies the complete address specified in that GET request before it is sent to the proxy cache. Specifically, the IP address of the origin server found in the destination field in the IP header of the one or more packets from the client containing the GET request is added by the proxy redirector as a prefix to the complete URL in the GET request to form an absolute URL. The proxy cache determines from that absolute URL whether it has the requested object stored in its cache. If it does, it sends the object back to the proxy redirector, which masquerades those packets as coming from the origin server by translating their destination address to the address of the client and their source address to that of the origin server. If the proxy does not have the requested object, a separate TCP connection is established between the proxy and the origin server from where the object is retrieved and then forwarded over the TCP connection between the client and the proxy. In order to account for the additional number of bytes in the GET request, an acknowledgement sequence number in packets returned 47 from the proxy that logically follow receipt of the GET request are decremented by that number by the proxy redirector before being forwarded to the client. Similarly, a sequence number in packets transmitted by the client subsequent to the GET request are incremented by that number before being forwarded by the proxy redirector to the proxy cache.

OPIC
Office de la propriété intellectuelle du Canada

CIPO
Canadian Intellectual Property Office

(12) (19) (CA) **Demande-Application**

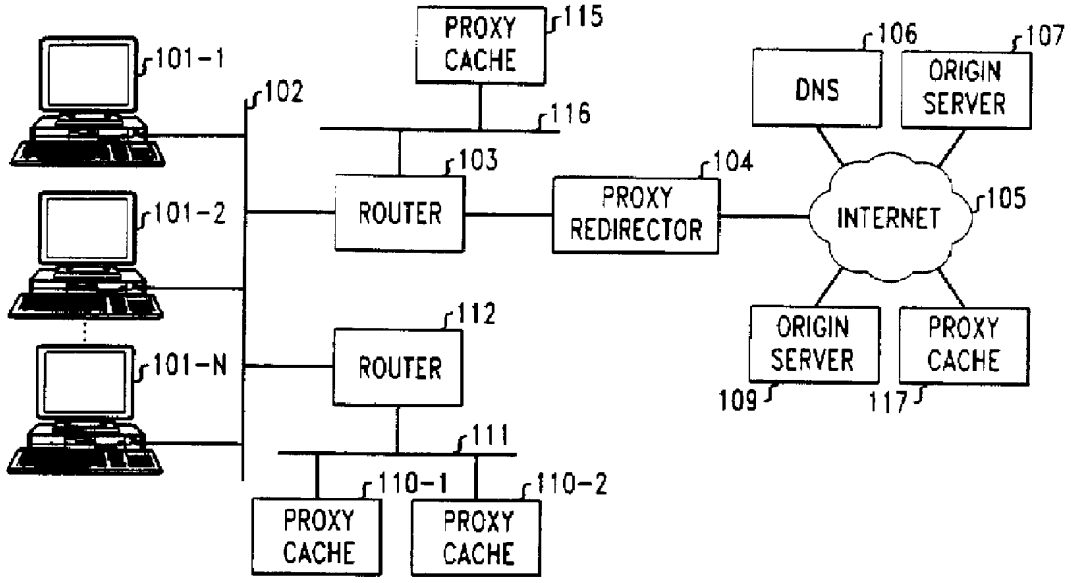
(21) (A1) **2,288,488**
(22) 1999/11/03
(43) 2000/06/16

(72) COHEN, ARIEL, US
(72) RANGARAJAN, SAMPATH, US
(72) SINGH, NAVJOT, US
(71) LUCENT TECHNOLOGIES INC., US

(51) Int.Cl.[7] H04L 29/06, H04L 12/56
(30) 1998/12/16 (09/212,980) US

(54) **METHODE ET DISPOSITIF D'ACHEMINEMENT TRANSPARENT DES DEMANDES D'OBJETS WEB VERS UN CACHE DE MANDATAIRE**

(54) **METHOD AND APPARATUS FOR TRANSPARENTLY DIRECTING REQUESTS FOR WEB OBJECTS TO PROXY CACHES**



(57) In order to transparently redirect an HTTP connection request that is directed to an origin server (107) to a proxy cache (110-1), a proxy redirector (104) translates the destination address of packets directed to the origin server to the address of the proxy. During a handshaking procedure, a TCP connection is transparently established between the client (110-1) and the proxy cache. When the client transmits a GET request to what it thinks is the origin server, which request specifies the complete address of an object at that origin server that it wants a copy of, the proxy redirector modifies the complete address specified in that GET request before it is sent to the proxy cache. Specifically, the IP address of the origin server found in the destination field in the IP header of the one or more packets from the client containing the GET request is added by the proxy redirector as a prefix to the complete URL in the GET request to form an absolute URL. The proxy cache determines from that absolute URL whether it has the requested object stored in its cache. If it does, it sends the object back to the proxy redirector, which masquerades those packets as coming

(21) (A1) **2,288,488**
(22) 1999/11/03
(43) 2000/06/16

from the origin server by translating their destination address to the address of the client and their source address to that of the origin server. If the proxy does not have the requested object, a separate TCP connection is established between the proxy and the origin server from where the object is retrieved and then forwarded over the TCP connection between the client and the proxy. In order to account for the additional number of bytes in the GET request, an acknowledgement sequence number in packets returned 47 from the proxy that logically follow receipt of the GET request are decremented by that number by the proxy redirector before being forwarded to the client. Similarly, a sequence number in packets transmitted by the client subsequent to the GET request are incremented by that number before being forwarded by the proxy redirector to the proxy cache.

METHOD AND APPARATUS FOR TRANSPARENTLY DIRECTING REQUESTS FOR WEB OBJECTS TO PROXY CACHES

Abstract of the Disclosure

In order to transparently redirect an HTTP connection request that is directed to an origin server (107) to a proxy cache (110-1), a proxy redirector (104) translates the destination address of packets directed to the origin server to the address of the proxy. During a handshaking procedure, a TCP connection is transparently established between the client (110-1) and the proxy cache. When the client transmits a GET request to what it thinks is the origin server, which request specifies the complete address of an object at that origin server that it wants a copy of, the proxy redirector modifies the complete address specified in that GET request before it is sent to the proxy cache. Specifically, the IP address of the origin server found in the destination field in the IP header of the one or more packets from the client containing the GET request is added by the proxy redirector as a prefix to the complete URL in the GET request to form an absolute URL. The proxy cache determines from that absolute URL whether it has the requested object stored in its cache. If it does, it sends the object back to the proxy redirector, which masquerades those packets as coming from the origin server by translating their destination address to the address of the client and their source address to that of the origin server. If the proxy does not have the requested object, a separate TCP connection is established between the proxy and the origin server from where the object is retrieved and then forwarded over the TCP connection between the client and the proxy. In order to account for the additional number of bytes in the GET request, an acknowledgement sequence number in packets returned

from the proxy that logically follow receipt of the GET request are decremented by that number by the proxy redirector before being forwarded to the client. Similarly, a sequence number in packets transmitted by the client subsequent to the GET request are incremented by that number before being forwarded by the proxy redirector to the proxy cache.

METHOD AND APPARATUS FOR TRANSPARENTLY DIRECTING REQUESTS FOR WEB OBJECTS TO PROXY CACHES

Field of the Invention

This invention relates to packet-switched computer networks, and more particularly, to a method and apparatus in such a network for transparently intercepting client web requests and redirecting them to proxy caches.

Background of the Invention

Proxy caching is currently used to decrease both the latency of object retrieval and traffic on the Internet backbone. As is well known, if a proxy cache has stored a copy of an object from an origin server that has been requested by a client, the requested object is supplied to the client from the proxy cache rather than from the origin server. This, therefore, obviates the need to send the request over a wide area network, such as the Internet, to the origin server where the original object is stored and the responsive transmission of a copy of the requested object back over the network to the requesting client.

Direction of a request from a client to a proxy cache to determine whether a requested copy of an object is stored in the cache can be accomplished either transparently or non-transparently to the client. Non-transparent redirection is accomplished through the client's browser program which is configured to send all object requests to a designated proxy cache at a specified address. Generally, a browser can be configured to send all of its client requests to a designated proxy cache if the client is

connected on a Local Area Network (LAN), or on an Intranet behind a firewall, where a proxy cache associated with that LAN or Intranet is located. When clients are served by a large Internet Service Provider (ISP), however, it is not advantageous from the ISP's standpoint to allow its subscribers to set their browsers to a specific proxy cache associated with the ISP. A large ISP likely will have many proxy caches in several locations and will thus want to maintain control over which of its several particular proxy caches a client request is directed. Further, if a proxy cache whose address is statically set in a client's browser becomes inoperative, all client requests will fail.

It is therefore more desirable from an ISP's standpoint with respect to latency and minimizing traffic onto and off of the network to transparently intercept a client's web request and send it to one of its operative proxy caches to determine whether a copy of the requested object is stored there. If a copy of the requested object is then found to be stored in that proxy cache, a copy of the object is sent to the client, which is unaware that it has been served an object from the proxy cache rather than from the origin server to which it made the request. If the proxy cache does not hold a copy of the requested object, then a separate connection is established between the proxy cache and the origin server to obtain a copy of the object, which when returned to the proxy is sent to the client over the connection established between the client and the proxy.

When a client specifies a URL of the object it is requesting a copy of, a Domain Name Server (DNS) look-up is performed to determine from the URL an IP address of an origin server which has that requested object. As a result of that look-up, an IP address is returned to the client of one of what may be several substantially equivalent servers that contain that object.

The client then establishes a TCP connection to that server using a three-way handshake mechanism. Such a connection is determined at each end by a port number and an IP address. First, a SYN packet is sent from the client to that origin server, wherein the destination IP address specified in the packet is the DNS-determined IP address of the origin server and the destination port number for an HTTP request is conventionally port 80. The source IP address and port number of the packet are the IP address and port number associated with the client. The client IP address is generally assigned to the client by an ISP and the client port number is dynamically assigned by the protocol stack in the client. The origin server then responds back to the client with an ACK SYN packet in which the destination IP address and destination port are the client's IP address and port number and the packet's source IP address and port number are the server's IP address and the server's port number, the latter generally being port 80. After receipt of the ACK SYN packet, the client sends one or more packets to the origin server, which packets include a GET request. The GET request includes a complete URL, which identifies to that server the specific object within the origin server site that the client wants a copy of. Unlike an absolute URL, which includes both site information (e.g., www.yahoo.com), and object information (e.g., index.html), a complete URL only identifies the particular object (e.g., index.html) that is requested since the packet(s) containing the GET request is sent to the proper origin server site by means of the destination address of the packet(s).

When a browser is configured to non-transparently send all requests to a proxy, a GET request is formulated by the browser that includes the absolute URL of the requested object. That absolute URL is then used by the proxy to establish a separate TCP connection to the origin server if the

proxy does not have a copy of the requested object in its cache. The proxy requires the absolute URL since the destination address of the packets to the proxy is set by the browser to the IP address of the proxy rather than the IP address of the origin server. Thus, in order to determine whether it has the object in its cache and if not establish a connection to the origin server, the proxy requires the absolute URL of the origin server in the GET request.

When requests are transparently directed to a proxy cache, however, the client browser is unaware that the request is being directed to the proxy and is possibly being fulfilled from the cache. Rather, the client's browser needs to "think" that it is connected to the origin server to which its SYN and the packet(s) containing the GET request are addressed. Such origin server IP address is determined by the browser through a DNS look-up. Further, the source address of the ACK SYN packet and the packets containing the requested object must be that same origin server IP address or they will not be recognized by the browser as being the responsive packets to the SYN packet and the request for the object. Thus, in order to transparently send object requests to a proxy cache, a mechanism must be in place along the packet transmission path to intercept an initial SYN packet sent by a browser and to redirect it to the proxy cache to establish a TCP connection. The proxy cache must then masquerade as the origin server when sending the ACK SYN packet back to the client by using the origin server's IP address and port number as the source address of that packet. Further, the subsequent packet(s) containing a GET request must be redirected to the proxy cache and the request fulfilled either from the cache or via a separate TCP connection from the proxy to the origin server. In either case, the source address of packets sent back to the client must be

the origin server's IP address and port number to which the packets sent by the client are addressed.

In order for packets associated with a request for an object to be redirected to a proxy cache connected somewhere in the network, a Layer 4 (L4) switch on the packet path "looks" at the port number of a destination address of a SYN request packet. Since HTTP connection requests are generally directed to port 80 of an origin server, the L4 switch transparently redirects all packets having a port number of 80 in the destination address. The SYN packet is thus sent to a selected proxy cache. In order for the proxy cache to properly respond to the client, as noted, it must know the absolute URL of the requested object and packets returned to the client must masquerade as coming from the origin server. Unlike the non-transparent caching method previously described in which the browser formulates a GET request with the absolute URL, for transparent caching the absolute URL must be provided in some manner to the proxy cache in order for the proxy to determine whether it in fact has the requested object in its cache, or whether it must establish a separate TCP connection to the origin server to request the object. In the prior art, when one or more caches are directly connected to the L4 switch, the switch chooses one of the caches and transparently forwards the packets to that proxy without modifying the source or destination address of the packets. The proxy, working in a promiscuous TCP mode accepts all incoming packets regardless of their destination address. The proxy, then receiving the SYN packet with the origin server's destination address and the client's source address, can respond to SYN packet with an ACK SYN packet. This ACK SYN packet has the client's address as a destination address and a source address masquerading as the origin server address. This packet is

transported through the L4 switch onto the network over the TCP connection back to the client. The subsequent packet(s) with the GET request from the client is redirected by the L4 switch to the directly connected proxy. Since the GET packet(s) only contains the complete URL, the proxy must formulate the absolute URL to determine whether its has the requested object in its cache or whether is must establish a separate TCP connection to the origin server. The proxy forms the absolute URL by prefixing the complete URL in the GET request with the IP address of the origin server in the destination address of the packet. The proxy can then determine whether it has the object and, if not, establish a TCP connection to that absolute address. If that particular origin server at that IP address should be inoperative, the proxy can alternatively prefix the complete URL in the GET request with the logical name of the site indicated in the HOST field in the packet(s) containing the GET request.

In the prior art, if the proxy cache is not directly connected to the L4 switch, then the L4 switch must perform a network address translation (NAT) and port address translation (PAT) on those packets directed to port 80 of an origin server. Specifically, when the L4 switch receives a SYN packet to initiate a TCP connection from a client to an origin server, it translates the destination address of the packet from the IP address and port number of the origin server to the IP address and port number of a selected proxy cache. Further, the switch translates the source address of the packet from the clients IP address and port number to its own IP address and a port number. When the proxy responds with an ACK SYN packet, it therefore responds to the L4 switch where a NAT translates the destination IP address from the IP address of the L4 switch to the IP address of the client, and translates the source IP address from the IP

address of the proxy to IP address of the origin server. A PAT also translates the port number in the destination address from that of the L4 switch to that of the client, and translates the port number in the source address from that of the proxy to that of the origin server (usually 80). When the client sends an ACK packet and then the packet(s) containing the GET request to the origin server, the L4 switch again performs a NAT, translating the destination IP address to the IP address of the proxy. Thus, when the packet(s) containing the GET request is received by the proxy, it does not know the IP address of the origin server as in the directly connected proxy arrangement described above. The proxy must therefore look at the logical name in the HOST field and perform a DNS look-up to determine that site's IP address. The proxy then uses that IP address in combination with the complete URL in the GET request to form an absolute URL from which it determines whether it has the requested object in its cache. If it doesn't, a separate TCP connection is established from the proxy to that absolute URL to retrieve that object, which is returned to the proxy. Whether the object is found in the proxy cache or is retrieved over the separate connection from the origin server, it is forwarded back to the L4 switch where a NAT and PAT are performed to translate the destination address to that of the client and to translate the source address to the particular origin server to which the client's request was directed. It should be noted that the source address of the origin server obtained when the client's browser initiates a DNS look-up using the origin server's absolute URL may not be the same IP address obtained when the proxy performs a DNS look-up using the combination of the site URL in the HOST field and the complete URL in the GET request.

The above described techniques for performing transparent proxy caching have several disadvantages. Firstly, use of a HOST field to specify a logical name of an origin server is not currently incorporated within the presently employed HTTP1.0 standards. Thus, a HOST field may not be present in the packet(s) containing a GET request. Where, as described above, the information in the HOST field is necessary to form an absolute URL to determine whether the proxy cache has the requested object and, if not, to establish a connection to an origin server from the proxy, the absence of the HOST field results in an unfilled request. Secondly, the prior art techniques require the proxy cache to perform the function of forming an absolute URL from the information in the HOST field and in the packet(s) containing the GET request. Thus, standard proxy caches which expect the client's browser to produce the absolute URL cannot be used. A methodology for transparent proxy caching that is transparent to both the client and the proxy is desirable to avoid modification to the program that controls proxy cache operations. Standard proxy caches could thus be employed anywhere in the network without the need for a special implementation.

The above described prior art techniques have even further disadvantages with respect to persistent connections defined by the HTTP1.1 standards. As defined by these standards, a persistent connection enables a client to send plural GET requests over the same TCP connection once that connection has been established between two endpoints. When a prior art transparent proxy cache is interposed on the connection, a client may "think" it has established a persistent connection to the specific origin server determined through the DNS look-up. The connection in reality, however, is transparently diverted by the L4 switch to a

proxy cache. The proxy cache, in response to a DNS look-up using the logical name in the HOST field, may be directed to an equivalent origin server at a different IP address. Further, as each subsequent GET request is received by the proxy from the client within the client's perceived persistent connection, each responsive DNS look-up to the logical name may direct a connection to an even different IP address of an equivalent origin server. As a result, the advantages of a transaction-oriented persistent connection in which a server is capable of maintaining state information throughout the connection, are lost. A methodology is desirable that maintains persistence to the same origin server to which the clients browser is directed, or to a same equivalent origin server throughout the duration of the persistent connection.

Summary of the Invention

The problems associated with the prior art techniques for transparent proxy caching are eliminated by the present invention. In accordance with the present invention, a switching entity, such as the L4 switch (referred to hereinafter as a proxy redirector), through which the packets flow, is provided with the functionalities at the IP level necessary to transform the complete URL in each GET request transmitted by a client to an appropriate absolute URL. Specifically, the IP address found in the destination field in the IP header of the packet(s) from the client containing the GET request are added as a prefix by the proxy redirector to the complete URL in the GET request. As a result, the complete URL in the GET request is modified to form an absolute URL which, when received by the proxy cache, is directly used to determine if the requested object is stored in the cache and, if not, to establish a separate TCP connection to the origin server. The GET

request received by the proxy is thus equivalent to what it would expect to receive if it were operating in the non-transparent mode. Advantageously, if a persistent connection is established, each subsequent GET request has the same IP address prefix determined by the initial DNS look-up by the client.

By modifying the GET request at the proxy redirector to include the destination address of the origin server, the number of bytes at the IP level in the packet containing the resultant absolute address are increased by the number of bytes in the prefix. Included in the header within each packet is a sequence number (seq) that provides an indication of the position of the first byte number in the payload. Thus, when the IP address is added to a packet, the sequence number of each of the subsequent packets needs to be incremented by the count of the added bytes. Further, an acknowledgement sequence number (ack_seq) in the header on the packets returned from the proxy or the origin server that logically follow receipt of the GET packet(s) at the origin server needs to be decremented by the proxy redirector before being forwarded to the client to avoid confusing the client with respect to what the sequence number of the next byte it sends should be. Further, if the GET request sent by the client encompasses more than one TCP segment, then the extra bytes in the first of the segments caused by the additional bytes added to the URL are shifted into the second segment, and the resultant now extra bytes in the second segment are shifted into the third segment, etc., until the last of the segments. In order to preclude the necessity of requiring an extra segment to be added to the GET request to accommodate the extra bytes, the client sending the GET request is deceived into sending segments whose maximum size is less than what can actually be received by the proxy as

indicated by a maximum segment size (MSS) field in packets from the proxy. The proxy redirector, upon receipt of the ACK SYN packet from the proxy, reduces the MSS parameter received from the proxy by the amount of the number of bytes that will be added to the GET request before that parameter is forwarded to the client. Thus, when the client next sends a GET request, each segment is limited to the reduced MSS, thereby insuring that the segment size of a last segment in a GET request after the IP address is prefixed by the proxy redirector to form the absolute URL (whether the GET request is one or more segments long) is less than or equal to the actual MSS that the proxy can receive.

Brief Description of the Drawing

FIG. 1 is a block diagram of a network that includes a proxy redirector that transparently sends requests from a client to a proxy cache by changing the destination address of packets in a client request from that of the origin server to that of a proxy and the source address from that of the client to that of the switching entity and, in accordance with the present invention, modifies a GET request to include the destination address of the origin server;

FIG. 2 is a block diagram showing the proxy redirector implemented on a programmable network element that manipulates packets in accordance with instructions provided by a loaded program; and

FIG. 3, 4, 5 and 6 are flow charts detailing the operation of the proxy redirector.

Detailed Description

With reference to FIG. 1, a plurality of clients 101-1 – 101-N are connected to a local area network (LAN) 102, such as an Ethernet. LAN 102, which, in turn, is connected through a router 103 to a Level 4 (L4) switch 104 (proxy redirector) which interfaces the LAN with a wide area network (WAN) 105, such as the Internet. Although shown as two separate elements, the functionalities of router 103 and proxy redirector 104 can in actual practice be combined in a single unit. All requests from any of the clients connected to LAN 102 for objects stored in servers connected to the Internet 105 traverse proxy redirector 104 onto the Internet. The packets comprising such requests, which include the standardized packets that establish a TCP connection, are directed to an IP destination address and port number indicated in the IP header of each packet originating from a client source address that includes a client IP address and port number. Similarly, responses to such requests from an origin server connected to Internet 105 are directed via an IP destination address that is equal to the client's IP address and port number from which the request originated, and have as a source address the server's IP address and port number. All such packets directed to any of the clients 101-1 – 101-N from any server connected to Internet 105 pass through proxy redirector 104.

When any of the clients connected to LAN 102, such as client 101-1, makes a request through a browser for an object by specifying a logical URL, a domain name server (DNS) 106 connected locally or on Internet 105, as shown, is accessed to perform a database look-up based on that logical name. An associated IP address is then returned to the browser. The IP address returned to the browser is the IP address of a particular origin server which contains the object requested through the logical URL. Since a logical name may in fact be associated with a plurality of essentially

equivalent origin servers, such as servers 107 and 109, the particular IP address returned to the client browser chosen by DNS 106 may be determined in a round-robin manner. When DNS 106 selects an origin server corresponding to the logical URL, the IP address of the selected origin server, such as, for example, the IP address of origin server 107, is returned to the browser in the requesting client 101-1. That IP address then serves as the IP address to which packets directed to the origin server from the client are directed. Conventionally, http requests are usually directed to port 80 of an origin server.

With the IP address of the origin server determined and returned to the client, the browser establishes a TCP connection between the client and the origin server through a three-way handshaking process. Specifically, a SYN packet, addressed to the IP address of the selected origin server, is sent by the client. Handshaking is completed when the client receives an acknowledgement of receipt of that SYN packet through an ACK SYN packet sent by that origin server, and responds with a ACK packet to the origin server. The browser then sends a GET request that specifies the particular requested object.

In accordance with the present invention, once the IP address of the origin server corresponding to the logical URL name is determined through the DNS look-up, proxy redirector 104, rather than establishing a TCP connection to the origin server at the determined IP address, transparently establishes a TCP connection between the client and a proxy. If the requested object is stored in the cache, a copy of that object is transparently returned to the requesting client. A TCP connection, therefore, is not established over the Internet 105, to the actual origin server 107 to provide the requested object to the requesting client. The coss of transmitting the

request to the origin server over the Internet and transmitting the copy of the requested object back over the Internet are thereby saved in addition to the time required for transmitting such a request over the Internet and waiting for a response from the origin server. If the proxy cache to which the request is directed does not contain the requested object, a separate TCP connection is established between the proxy cache and the origin server to obtain a copy of the requested object. When the proxy cache then receives the copy of the requested object from an origin server over that separate TCP connection, the copy is forwarded to the client over the original TCP connection that was established between the client and the proxy cache.

In the embodiment shown in FIG. 1, a proxy cache 110-1 is illustratively shown connected to a LAN 111, which is connected to LAN 102 through a router 112. Another proxy cache 115 is shown connected on a different LAN 116 through router 103. Other proxy caches can be located anywhere on LANs 102, 111, or 116, on another LAN connected to the Internet 105 such as proxy cache 117. Proxy redirector 104 selects one of the available proxy caches to which to forward client requests based on a metric such as least-loaded or round-robin, based on IP header information such as the origin server IP address. With respect to the latter, all objects from a specific origin server will be served by a specific proxy.

In the preferred embodiment described herein, proxy redirector 104 includes a programmable network element of the type described in co-pending U.S. patent application Serial Number 09/190,355, filed November 12, 1998, which application is incorporated herein by reference. As described in that application, that programmable network element in the preferred embodiment runs on a Linux machine. As shown in FIG. 2, the programmable network element 200 includes a dispatcher process 202 with

which plural different gateway programs (204, 205 and 206) register and request access to IP packets that fit specific descriptions. Such programs are loaded through an admission daemon 210 via a local program injector 211 or a remote program injector 212. A gateway program, for example, can request access to incoming packets to network interface 208 that match certain source and destination IP address ranges and port numbers. The dispatcher process 202 uses a packet filter 203 in the Linux kernel 201 to obtain packets requested by the gateway programs and uses a raw IP socket 215 to send packets that have been manipulated in accordance with the gateway program back to the kernel for output back to the network through filter 203 through network interfaces 208. Library functions are provided in the programmable network element that enable a gateway program to communicate with the dispatcher process 202 to register rules that specify the type of IP packets that a gateway program wants diverted to it. A gateway program can request either a complete IP packet or only the IP and TCP header of a packet and can change both the header and payload of a packet.

In the present invention, that programmable network element is operative in combination with a gateway program that manipulates the destination and source addresses of packets flowing there through in a manner to be described, as well as modifying, as will be described, information in the packet(s) containing the GET request that specifies the URL of the requested object. Specifically, the programmable network element in combination with the gateway program operates on packets associated with HTTP requests, which are determined from the destination port number. As previously noted, HTTP requests are conventionally addressed to port 80 of an origin server. Thus, the programmable network

element/gateway program which together comprise proxy redirector 104 in this embodiment, captures through the dispatcher process of the programmable network element, packets directed to port 80 and then performs address translations on those captured packets to readdress these packets to a selected proxy. With respect to address translations, the gateway program translates the destination IP address of packets addressed to the origin server to the IP address of a selected proxy cache and translates the source IP address of such packets from that of the client to the IP address of proxy redirector 104. Further, in order for proxy redirector 104 to identify requests from plural client terminals that are directed to the same proxy, the source port number is translated to a bogus ghost port number at the proxy redirector. Thus, when proxy cache responds, the packets transmitted by the cache have a destination IP address of proxy redirector 104 at that bogus port number, which is distinctly associated with the client. The gateway program within proxy redirector 104 then translates the IP destination address of these responsive packets from the proxy to the IP address of the client and translates the bogus destination port number to the port number from which the client originated its request. Further, the gateway program translates the source IP address of such responsive packets from that of the proxy to the IP address of the origin server and the port number to the port (80) to which the client's requests were originally directed. Thus, the packets which are returned to the client from the proxy masquerade as if they had originated from the origin server to which the client "believed" its request had been sent.

By performing the above-described network address translations (NATs) and port address translations (PATs), packets from a client 101-1

are transparently directed by proxy redirector 104 to a proxy cache. Responsive packets from the proxy cache are sent to proxy redirector 104 where they are redirected to client 101-1.

In establishing a TCP connection that is directed to an origin server, client 101-1 first transmits a SYN packet, which is intercepted by proxy redirector 104. Proxy redirector 104 selects a proxy cache, such as proxy 110-1, to redirect this request and creates a connection control block (CCB) to maintain information about the connection. Selection of the particular proxy is determined, as described above, by one of several possible algorithms. The CCB is used to store the client IP address and TCP port number and the origin server IP address and TCP port number, both of which are contained in the IP header of the SYN packet, and the chosen proxy's IP address. The destination address is then changed to that of the chosen proxy and the packet is sent back to the network for redirection to its new destination address of the proxy 110-1. All subsequent packets that originate from the same client with the same TCP port number are then forwarded to the same proxy. Proxy 110-1 responds with an ACK SYN packet which is directed via its destination address to proxy redirector 110-1. Proxy redirector 104 then translates the source IP address and port number to those of the origin server and the destination IP address and port number to those of the client. When the packet arrives at the client the client believes that it is connected to the origin server. The client then responds with an ACK packet to the origin server, which is redirected by proxy redirector 104 to proxy cache 110-1, to complete the handshaking process.

After the TCP connection is established between client 101-1 and proxy cache 110-1, client 101-1 sends one or more packets containing a

GET request addressed to the origin server. Such packets are thus "captured" by proxy redirector 104 and redirected to proxy cache 110-1. As previously discussed, the GET request sent by the client contains only the complete URL sent by the client browser which in itself provides insufficient information for the proxy cache to determine whether it has the requested object and, if not, to forward it to the origin serve,r which does. In accordance with the present invention, the gateway program that is operative with the programmable network element of the proxy redirector 104, captures this packet or packets and, in addition to previously described address translations, transforms the complete URL to an absolute URL by prefixing it with the IP address of the origin server obtained from the destination IP address of the packet(s) containing the GET request. Thus, Level 7 (application) information is modified to assist in level 4 routing.

In order to make the URL transformation transparent to both the client and proxy cache endpoints, changes in IP and TCP headers are also required. Since the GET request modification increases the length of the IP packet that carries the GET request, the total length field on the IP header of this packet is increased by an offset. The offset amount is recorded in the CCB. In addition, the TCP header contains sequence numbers (seq) and acknowledgement sequence numbers (ack_seq) that need to be translated. The seq in the TCP header indicates the byte number of the first byte on this packet going from the sender to the receiver over the TCP session and the ack_seq indicates the byte number of the next byte that the sender expects to receive from the receiver. For all packets after the GET packet(s) that go from the client to the proxy cache, the seq is increased by an offset equal to the *lengthof*(absolute URL) – *lengthof*(complete URL) so that the seq matches the byte number of the byte that the proxy cache

expects to receive from the client. Similarly, on all packets starting with the acknowledgement to the GET packet that go from the proxy cache to the client through the proxy redirector, the ack_seq is decreased by the same offset so that the ack_seq matches the byte number of the byte that the proxy cache would expect the client to send in the next packet following the GET packet. By performing these changes in the header, the client and proxy cache endpoints remain unaware of the modification in the GET packets from the complete URL to the absolute URL.

Table 1 illustrates the URL and other header transformations performed

Table 1

Arriving packet:

---> beginning of packet header dump <---
---> IP header: version=4 hdr_len=5 TOS=0 pkt_len=346 id=60276
frag_off=4000H TTL=64 protocol=6 cksum=1792H
saddr=135.104.25.243 daddr=204.71.200.244
---> TCP header: sport=1273 dport=80 seq=2189084427
ack_seq=3266449517
tcp_hdr_len=5 flags=ACK PSH
res1=0H res2=0H window=31856 cksum=162H urgent=0
---> beginning of packet data dump <---
GET /a/ya/yahoomail/promo1.gif HTTP/1.0
Referer: http://www.yahoo.com/
Connection: Keep-Alive
User-Agent: Mozilla/4.05 [en] (X11; U; Linux 2.1.103 i686)
Host: us.yimg.com
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg image/png
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8

Modified to:

```
---> beginning of packet header dump <---
---> IP header: version=4 hdr_len=5 TOS=0 pkt_len=367 id=60276
    frag_off=4000H TTL=64 protocol=6 cksum=1792H
    saddr=135.104.25.245 daddr=135.104.25.31
---> TCP header: sport=5000 dport=7000 seq=2189084427
ack_seq=3266449517
    tcp_hdr_len=5 flags=ACK PSH
    res1=0H res2=0H window=31856 cksum=162H urgent=0
---> beginning of packet data dump <---
GET http://204.71.200.244/a/ya/yahoomail/promo1.gif HTTP/1.0
Referer: http://www.yahoo.com/
Connection: Keep-Alive
User-Agent: Mozilla/4.05 [en] (X11; U; Linux 2.1.103 i686)
Host: us.yimg.com
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg image/png
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
```

---

on a GET packet that arrives at proxy redirector 104 from a client 101-1. The packet is destined to an origin server at a destination IP address (daddr) *204.71.200.244* (www.yahoo.com) at a destination port (dport) *80*, requesting object */a/ya/yahoomail/promo1.gif*. As can be noted in the modified packet, the packet is redirected to a proxy cache 110-1 at IP address *135.104.25.31* port *7000* by changing the daddr and dport header information. Also, the complete URL of the object in the GET request is modified in the translated packet by prefixing it with *http://204.71.200.244* to form the absolute URL, where that prefix is obtained from the daddr header in the arriving packet. This transformation increases the length of the packet by 21 bytes so that the pkt_len field in the header is modified from *346* to *367* bytes. Further, the source port is modified to a bogus port number by changing sport to *5000*.

Table 2 shows the translations performed by the proxy redirector 104 to an acknowledgment from proxy cache 110-1 to the GET request. The arriving

Table 2

Arriving packet:

---> beginning of packet header dump <---
---> IP header: version=4 hdr_len=5 TOS=0 pkt_len=40 id=14559
frag_off=4000H TTL=255 protocol=6 cksum=10eH
saddr=135.104.25.31 daddr=135.104.25.245
---> TCP header: sport=7000 dport=5000 seq=3266449517
ack_seq=2189084754
tcp_hdr_len=5 flags=ACK
res1=0H res2=0H window=8433 cksum=32H urgent=0

Modified to:

---> beginning of packet header dump <---
---> IP header: version=4 hdr_len=5 TOS=0 pkt_len=40 id=14559
frag_off=4000H TTL=255 protocol=6 cksum=10eH
saddr=204.71.200.244 daddr=135.104.25.243
---> TCP header: sport=80 dport=1273 seq=3266449517
ack_seq=2189084733
tcp_hdr_len=5 flags=ACK
res1=0H res2=0H window=8433 cksum=32H urgent=0

packet is addressed (daddr) to the IP address of the proxy redirector at the bogus port (dport) *5000* used by the proxy redirector for this TCP connection. The source IP address (saddr) and the source port (sport) are those of the proxy cache to where the GET request was directed. The ack_seq field indicates the byte number of the first byte that is expected to be sent in the packet following the GET packet. In this example, ack_seq is equal to the sequence number (seq) *218908427* of the GET packet plus the length of the GET packet, which in this case per Table 1 is *367*. Thus ack_seq of the arriving packet is *218908754*. Since client 101-1 is unaware

that a proxy redirector has increased the length of the GET packet by 21 bytes, proxy redirector 104 decreases the ack_seq field by the number of bytes added, 21. Further, proxy redirector 104 translates the destination IP address and port number to those of the client 101-1, and the source IP address and port number to that of the origin server. The modified packet, when received by the client, thus appears to the client to have originated from the origin server and the ack_seq field indicates a byte number that the client would next expect to send having previously sent a packet of length 367 bytes. All packets that are subsequently sent through the proxy redirector 104 to client 101-1 from proxy cache 110-1 are similarly modified to decrement the ack_seq field by the number of bytes, 21, added to the GET packet.

Table 3 illustrates a next packet sent by client 101-1 to the origin server after the GET packet. In this packet the sequence number (seq) is equal to the modified ack_seq sent to the client, as per Table 2. The destination IP address and port number are those of the origin server and the source IP address and port number are those of the client. When received by proxy redirector 104, the packet is modified to change the source IP address and port number to the IP address and bogus port number of the proxy redirector. The destination address IP and port number are translated to that of proxy cache 110-1. The sequence number (seq) is increased by that same value of 21 to match the byte number that the proxy cache expects to receive based on the sequence number previously received in the GET packet and the length, 367, of the GET packet

Table 3

Arriving packet:

---> beginning of packet header dump <---
---> IP header: version=4 hdr_len=5 TOS=0 pkt_len=40 id=60281
frag_off=4000H TTL=64 protocol=6 cksum=18bfH
saddr=135.104.25.243 daddr=204.71.200.244
---> TCP header: sport=1273 dport=80 seq=2189084733
ack_seq=3266450977
tcp_hdr_len=5 flags=ACK
res1=0H res2=0H window=31856 cksum=d3f7H urgent=0

Modified to:

---> beginning of packet header dump <---
---> IP header: version=4 hdr_len=5 TOS=0 pkt_len=40 id=60281
frag_off=4000H TTL=64 protocol=6 cksum=18bfH
saddr=135.104.25.245 daddr=135.104.25.31
---> TCP header: sport=5000 dport=7000 seq=2189084754
ack_seq=3266450977
tcp_hdr_len=5 flags=ACK
res1=0H res2=0H window=31856 cksum=d3f7H urgent=0

---

received. All subsequent packets directed to the origin server from client 101-1 are similarly modified before being directed to proxy cache 110-1.

In the above-description, it has been assumed that the length of the GET request both before modification, and after the URL extension is less than the maximum TCP segment size. In fact, the length of the GET request may be longer than one TCP segment. If the length of the GET request carrying the complete URL occupies *x* number of TCP segment and, after it is modified to carry the absolute URL, it still also fits into that same *x* number of TCP segments, then the segment carrying the URL is modified and overflowing characters are pipelined from one segment to the next. Thus, the overflowing characters from a previous packet are prefixed

to the start of the next packet, etc., until the last packet, which length is increased by the increased number of bytes due to the URL modification. Therefore, the packet length of only the last segment is modified to include the characters that have been shifted into that segment. The ack_seq parameter in packets from the proxy cache to the client is modified starting from the acknowledgment to the last GET packet.

If the modification of the URL to the absolute URL could cause the last TCP segment of the GET request to overflow to another segment, a new TCP segment would need to be constructed and injected by the proxy redirector. The proxy redirector would then need to have the capability to retransmit this segment if it was lost. Thus, the proxy redirector would need to have some TCP layer functionalities. In order to avoid adding higher level functionality to the proxy redirector, segment sizes are limited to less than what the proxy cache is actually capable of receiving. When the complete URL is transformed to an extended URL, the maximum increase in size is 22 bytes, equal to the maximum length of an IP address of 15 bytes plus 7 bytes from the prefix: http://. The client is deceived to send segments whose maximum size is 22 bytes less than what the protocol allows it to send. The TCP segment size sent by the client is determined by what the proxy cache, in its handshake with the client, indicates as the maximum segment size it can receive. This is indicates by the proxy cache through the maximum segment size (MSS) field in the ACK SYN packet. Accordingly, the proxy redirector 104 intercepts the ACK SYN packets and decreases the specified MSS amount by 22. For example, if the MSS specified by the proxy cache is 1460, it is modified to 1438 by the proxy redirector before being sent to the client. When the client next sends a GET request, the TCP segments are limited to 1438 bytes. In the worst case,

when the client sends a GET request, 22 bytes will be added to the *x*th TCP segment that carries this request. The length of this *x*th TCP segment will still then be within the maximum length specified by the proxy cache. If the event that the proxy cache does not stipulate a maximum MSS in the ACK SYN packet, the default used by the client is 536 bytes. An MSS option is then added by the proxy redirector to inform the client that the maximum MSS expected by the other end of the TCP connection is 514 bytes.

As previously described, a NAT and PAT are performed by proxy redirector 104 on all packets addressed by client 101-1 to an origin server, and all packets addressed by proxy cache 110-1 to proxy redirector 104 for return to the client. Proxy redirector 104 thus performs a NAT and a PAT on these packets flowing in both direction. If proxy redirector 104 selects a proxy cache that is located in such a point on the network that packets from the proxy cache addressed directly to client 101-1 must pass through proxy redirector 104 due to the network configuration, then proxy redirector need only perform a half-NAT on the packets flowing through it. Specifically, if proxy redirector 104 selects a proxy cache such as proxy cache 117, all packets addressed to client 101-1 must pass through proxy redirector 104. Proxy redirector 104 thus only needs to transform the destination IP address and port number of packets from client 101-1 to the IP address and port number of proxy cache 117, while maintaining the source IP address and port number as those of client 101-1. The packets returned from proxy cache 117 will thus be addressed to the client's IP address and port number. When they pass through proxy redirector 104, they are captured and the transformation of the source IP address and port to those of the origin server are the only address changes that need to be effected.

The problems of the prior art with respect to persistent connections is obviated in accordance with the present invention. As previously noted, during a persistent connection plural GET requests can be made by a client. In the prior art, as described, each GET request can result in a connection from a proxy cache to a different origin server if the proxy cache does not have the requested objects. The ability of a server to maintain the state of a client's connection throughout the duration of the connection is compromised if each GET request results in connections to multiple servers. In accordance with the present invention, once the IP address of the origin server is determined at the initial DNS lookup, that same IP address is used by the proxy redirector as a prefix to each complete URL in every GET request issued by the client throughout the duration of the persistent connection. Thus, assuming the proxy cache does not contain any of the requested objects, the proxy cache will establish a TCP connection to the same origin server in response to each GET request generated by the client. It should be noted that if plural client GET requests are forwarded by the proxy redirector to a proxy cache within a persistent TCP connection, ack_seq parameter in packets that flow through the proxy redirector from the proxy following each GET request must reflect the cumulative changes effected by translating the complete URL to the absolute URL in each of the preceding GET requests within the same TCP connection. Similarly, in all packets received by the proxy redirector from the client directed to the origin server within a persistent TCP connection, the seq parameter must reflect cumulative changes.

FIGS. 3, and 4, together are flow charts detailing the functions of proxy redirector 104 in establishing a TCP connection to a proxy cache and modifying the GET request so that such requests can be transparently

forwarded to the proxy. At step 301, a SYN packet arrives from the client at the proxy redirector At step 302, proxy redirector selects a proxy cache based on a load balancing algorithm or on an arbitrary or random selection. At step 303, proxy redirector performs a full NAT, changing the daddr from that of the origin server to that of the selected proxy and saddr from that of the client to that of the proxy redirector. At step 304 a PAT is performed, changing sport to that of a bogus ghost port number and dport to the proxy's port number. At step 305, the SYN packet is sent to the proxy. In response to that SYN packet, the proxy responds, at step 306, with a SYN ACK packet containing an MSS parameter in the TCP header. At step 307, a reverse translation is performed on both the IP addresses and port numbers, changing saddr and sport to those of the origin server and daddr and dport to those of the client. At step 308, the MSS field is changed by reducing the value of the MSS received from the proxy by 22. At step 309, the ACK SYN packet is sent to the client. At step 310, proxy redirector receives a responsive ACK packet from the client. At step 311, a full NAT and PAT are performed on that packet and, at step 312, the modified packet is sent to the proxy, thereby completing the handshake sequence.

At step 313, a packet containing a GET request is received from the client. A full NAT is performed at step 314 and a PAT is performed at step 315. A determination is made at decision step 316 whether this is a first packet in the GET request. If yes, at step 317, the IP address of the origin server obtained from daddr of the arriving packet is prefixed to the complete URL in the GET request. If, at step 316, the packet is not a first packet in a GET request, then, at step 318, the overflow bytes from the previous GET packet are prefixed to those bytes in the current packet and if the total number of bytes in the resultant packet is than the actual MSS sent by the

proxy, the overflow bytes greater are buffered for prefixing to the next packet. After alternative steps 316 or 318, at step 319, a determination is made whether the current packet is the last packet of a GET request. If not, at step 320, the current packet is sent to the proxy and the flow returns to step 313 to receive the next packet in the GET request from the client. If at step 319, the current packet is the last packet in a GET message, then, at step 321, the pkt_len parameter of that packet is changed to reflect the change in length of the packet. At step 322, the modified packet is sent to the proxy.

FIG. 5 illustrates the steps performed by the proxy redirector for each packet received from the proxy starting from the ACK to the GET request through the end of the connection. At step 501, the proxy redirector receives the ACK to the GET request, or any other packet that logically follows the ACK to the GET request. At step 502, reverse NAT and PATs are performed, translating daddr and dport to those of the client and saddr and sport to those of the origin server. At step 503, ack_seq is decreased by the amount added in the preceding GET request. At step 504, the modified packet is sent to the client.

FIG. 6 illustrates the step performed by the proxy redirector for each packet destined for the origin server from the client that follows the GET request. At step 601, a packet subsequent to the GET request is received from the client. At step 602, a full NAT and PAT are performed. At step 603, seq_no is increased by the amount of bytes added by modifying the previous GET request. At step 604, the packet is sent to the proxy.

In the discussion above of FIGS. 3, 4, 5 and 6, it has been assumed that the proxy cache is located in a position such that packets directed to the client will not automatically flow through the proxy redirector. Thus, all

packets from the proxy are addressed to the proxy redirector. Therefore, for packets flowing from the client, and packets flowing from the proxy, the proxy redirector performs a full NAT and PAT. If however, as previously described, the proxy cache selected by the proxy redirector is located on the network so that all packets from proxy to the client automatically flow through the proxy redirector, then, in the steps shown in FIGS. 3, 4, 5 and 6, only a half NAT needs to be performed.

Although described in conjunction the programmable network element shown in FIG. 2, the proxy redirector of the present invention could be implemented through other means, using hardware, software, or a combination of both. As an example, a level 4 switch having a fixed program to perform the required packet manipulations required by the present invention could be used.

As described, the proxy cache returns requested objects to the address from which a request originated as indicated by the saddr and sport parameters in the IP header information, which is the address of the proxy redirector 104 when the proxy cache is not connected on the network so that all responses do not automatically pass through the proxy redirector. The interactions between the requesting client and the proxy cache are transparent to both the client and the proxy cache, since the client does not "know" that its request is being redirected to the proxy by the proxy redirector, and the proxy cache, when receiving a GET request with an absolute URL does not know that that absolute URL is not being formulated by the client's browser operating in a non-transparent mode.

Advantageously, the proxy cache requires no software modifications and standard proxy caches, connected anywhere on the network can be used in conjunction with the proxy redirector. If, however, the proxy is modified,

using a programmable network element as previously described, for example, the requested object retrieved by the proxy from its own cache or received from an origin server, can be sent directly back to the client, thereby obviating the need to send such packets back to the proxy redirector for address translations and redirection to the client. By performing only a half-NAT at the proxy redirector and leaving the client's saddr and sport as the source IP address and port number in the header of the SYN packet, GET request packet(s), and other packets forwarded by proxy redirector 104 from the client, the proxy cache can return packets responsive to the request directly to the client by substituting the origin server's IP address and port number as the source address for its own address. If the proxy redirector performs a full NAT and PAT, then another mechanism must be incorporated to provide the client address to the proxy cache, such as incorporating the client address information as part of an appendix to the absolute address in the modified GET request and stripping the appended client address information at the proxy before determining whether the requested object is stored in the cache or whether a connection to the origin server need be made. Advantageously, by sending the packets from the proxy cache directly back to the client, the delay encountered by transmitting such packets back to the proxy redirector for address translation and redirection is eliminated. Disadvantageously, the proxy cache must be modified to perform these functions, precluding use of standard available proxy caches.

Although described hereinabove in connection with GET requests, the present invention can equally be applied to redirection of any type of request message in which the token is, for example, GET, POST or HEAD, or any other type of token yet to be implemented and/or standardized.

The foregoing therefore merely illustrates the principles of the invention. It will thus be appreciated that those skilled in the art will be able to devise various arrangements which, although not explicitly described or shown herein, embody the principles of the invention and are included within its spirit and scope. Furthermore, all examples and conditional language recited hereinabove are principally intended expressly to be only for pedagogical purposes to aid the reader in understanding the principles of the invention and the concepts contributed by the inventors to furthering the art, and are to be construed as being without limitation to such specifically recited examples and conditions. Moreover, all statements hereinabove reciting principles, aspects, and embodiments of the invention, as well as specific examples thereof, are intended to encompass both structural and functional equivalents thereof. Additionally, it is intended that such equivalents include both currently known equivalents as well as equivalents developed in the future, i.e., any elements developed that perform the same function, regardless of structure.

Thus, for example, it will be appreciated by those skilled in the art that the block diagrams and flowcharts described hereinabove represent conceptual views of illustrative circuitry and processes embodying the principles of the invention. Similarly, it will be appreciated that any flowcharts, flow diagrams, state transition diagrams, pseudocode, and the like represent various processes which may be substantially represented in computer readable medium and so executed by a computer or processor, whether or not such a computer or processor is explicitly shown.

The functions of the various elements shown in the FIGS., including functional blocks labeled as "processors" may be provided through the use of dedicated hardware as well as hardware capable of executing software in

association with appropriate software. When provided by a processor, the functions may be provided by a single dedicated processor, by a single shared processor, or by a plurality of individual processors, some of which may be shared. Moreover, explicit use of the term "processor" or "controller" should not be construed to refer exclusively to hardware capable of executing software, and may implicitly include, without limitation, digital signal processor (DSP) hardware, read-only memory (ROM) for storing software, random access memory (RAM), and non-volatile storage. Other hardware, conventional and/or custom, may also be included. Similarly, any switches shown in the FIGS. are conceptual only. Their function may be carried out through the operation of program logic, through dedicated logic, through the interaction of program control and dedicated logic, or even manually, the particular technique being selectable by the implementor as more specifically understood from the context.

In the claims hereof any element expressed as a means for performing a specified function is intended to encompass any way of performing that function including, for example, a) a combination of circuit elements which performs that function or b) software in any form, including, therefore, firmware, microcode or the like, combined with appropriate circuitry for executing that software to perform the function. The invention as defined by such claims resides in the fact that the functionalities provided by the various recited means are combined and brought together in the manner which the claims call for. Applicant thus regards any means which can provide those functionalities as equivalent to those shown hereinabove.

Claims

1. A method at a Layer 4 switch for transparently redirecting an HTTP connection request that is directed to an origin server to a proxy cache over a packet-based computer network comprising the steps of:

receiving at least one packet containing a request message within the HTTP connection, the at least one packet having an IP header comprising a destination address of an origin server, the request message including a complete address of a specified object at the origin server;

modifying the request message by combining the destination address of the origin server with the complete address; and

forwarding the at least one packet containing the modified request message to the proxy cache over the packet-based network.

2. The method of claim 1 wherein the step of modifying comprises the step of prefixing the complete address in the request message with the destination address of the origin server.

3. The method of claim 2 further comprising the step of:

translating the destination address in an IP header of the at least one packet containing the modified request message to the address of the proxy cache.

4. The method of claim 3 further comprising the step of:

translating a source address in the IP header of the at least one packet containing the modified request message from an address of the client to an address of the Layer 4 switch.

5. The method of claim 3 further comprising the step of:

in a TCP header in packets received from the proxy cache commencing with an acknowledgment to the modified request message, modifying a value in a acknowledged byte sequence number field to reflect the increased number of bytes in the modified request message resulting from the step of prefixing the complete address with the destination address of the origin server.

6. The method of claim 5 wherein the step of modifying the value in the acknowledged byte sequence number field comprises the step of decreasing the value in the acknowledged byte sequence number field by the number of bytes added by prefixing the destination address of the origin server to the complete address in the request message.

7. The method of claim 6 further comprising the step of:

translating in the IP header the destination address to the address of the client and the source address to that of the origin server in the packets received from the proxy cache commencing with the acknowledgment to the request message.

8. The method of claim 3 further comprising the step of:

modifying a value in a sent byte sequence number field to reflect the increased number of bytes in the modified request message resulting from prefixing the complete address with the destination address of the origin server in a TCP header in packets received from the client following receipt of the request message.

9. The method of claim 8 wherein the step of modifying the value in the sent byte sequence number field comprises the step of increasing the

value in the sent byte sequence number field by the number of bytes added by prefixing the destination address of the origin server to the complete address in the request message.

10. The method of claim 9 further comprising the step of:

translating the destination address to the address of the proxy cache in the IP header in each packet received from the client following receipt of the request message.

11. The method of claim of claim 1 further comprising the step of selecting the proxy cache from a plurality of different proxy caches prior to receiving the at least one packet containing the request message.

12. The method of claim 2 further comprising, prior to the step of receiving the at least one packet containing the request message, the steps of:

receiving a packet from the proxy cache indicating a maximum segment size that packets sent to it thereafter should have;

reducing by a predetermined number the maximum segment size of packets to be thereafter sent to the proxy cache; and

sending a packet to the client indicating the reduced maximum segment size that packets thereafter sent by the client to the origin server should have.

13. The method of claim 12 wherein the predetermined number is equal at least to the maximum number of bytes added by the step of prefixing the destination address to the complete address in the request message.

14. The method of claim 13 further comprising the steps of:

successively shifting to a next packet in the request message overflow bytes caused by the step of prefixing the destination address to the complete address in the request message, if the request message is contained within more than one packet; and

changing a length-of-packet parameter in the IP header in a last packet of a multi-packet request message to reflect the bytes shifted into the last packet from a previous packet.

15. The method of claim 13 further comprising the step of changing a length-of-packet parameter in the IP header to reflect the change in the length of that packet caused by the step of prefixing the destination address to the complete address in the request message, if the request message is contained within one packet.

16. The method of claim 1 wherein the request message is a GET request.

17. The method of claim 1 wherein the request message is a POST request.

18. The method of claim 1 wherein the request message is a HEAD request.

19. A proxy redirector for transparently redirecting an HTTP connection request from a client that is directed to an origin server request message to a proxy cache over a packet-based computer network comprising:

means for receiving at least one packet containing the request message, the at least one packet having an IP header comprising a

destination address of an origin server, the request message including a complete address of a specified object at the origin server;

means for modifying the request message by combining the destination address of the origin server with the complete address; and

means for forwarding the at least one packet containing the modified request message to the proxy cache over the packet-based network.

20. The proxy redirector of claim 19 wherein the means for modifying comprises means for prefixing the complete address in the request message with the destination address of the origin server.

21. The proxy redirector of claim 20 further comprising:

means for translating the destination address in an IP header of the at least one packet containing the modified request message to the address of the proxy cache.

22. The proxy redirector of claim 21 further comprising:

means for translating a source address in the IP header of the at least one packet containing the modified request message from an address of the client to an address of the proxy redirector.

23. The proxy redirector of claim 21 further comprising:

means for modifying a value in an acknowledged byte sequence number field in a TCP header in packets received from the proxy cache, commencing with an acknowledgment to the modified request message, to reflect the increased number of bytes in the modified request message resulting from prefixing the complete address with the destination address of the origin server.

24. The proxy redirector of claim 23 wherein the means for modifying the value in the acknowledged byte sequence number field decreases the value in the acknowledged byte sequence number field by the number of bytes added by prefixing the destination address of the origin server to the complete address in the request message.

25. The proxy redirector of claim 24 further comprising:

means for translating in the IP header the destination address to the address of the client and the source address to that of the origin server in the packets received from the proxy cache commencing with the acknowledgment to the request message.

26. The proxy redirector of claim 21 further comprising:

means for modifying a value in a sent byte sequence number field to reflect the increased number of bytes in the modified request message resulting from prefixing the complete address with the destination address of the origin server in a TCP header in packets received from the client following receipt of the request message.

27. The proxy redirector of claim 26 wherein the means for modifying the value in the sent byte sequence number field increases the value in the sent byte sequence number field by the number of bytes added by prefixing the destination address of the origin server to the complete address in the request message.

28. The proxy redirector of claim 27 further comprising:

means for translating the destination address to the address of the proxy cache in the IP header each packet received from the client following receipt of the Request message.

29. The proxy redirector of claim 19 further comprising means for selecting the proxy cache from a plurality of different proxy caches.

30. The proxy redirector of claim 20 further comprising:

means for receiving a packet from the proxy cache indicating a maximum segment size that packets sent to it thereafter should have;

means for reducing by a predetermined number the maximum segment size of packets to be thereafter sent to the proxy cache; and

means for sending a packet to the client indicating the reduced maximum segment size that packets thereafter sent by the client to the origin server should have.

31. The proxy redirector of claim 30 wherein the predetermined number is equal at least to the maximum number of bytes added by the prefix of the destination address to the complete address in the request message.

32. The proxy redirector of claim 31 further comprising:

means for successively shifting to a next packet in the request message overflow bytes caused by the prefix of the destination address to the complete address in the request message, if the request message is contained within more than one packet; and

means for changing a length-of-packet parameter in the IP header in a last packet of a multi-packet request message to reflect the bytes shifted into the last packet from a previous packet.

33. The proxy redirector of claim 31 wherein the proxy cache further comprises means for changing a length-of-packet parameter in the IP

header to reflect the change in the length of that packet caused by the prefix of the destination address to the complete address in the request message, if the length of the request message is contained within one packet.

34. The proxy redirector of claim 19 wherein the means for modifying the request message is a gateway program dynamically loaded on a programmable network element.

35. The proxy redirector of claim 19 wherein the request message is a GET request.

36. The proxy redirector of claim 19 wherein the request message is a POST request.

37. The proxy redirector of claim 19 wherein the request message is a HEAD request.

38. A computer readable medium storing computer program instructions which are executable on a computer system implementing a Layer 4 switch for transparently redirecting an HTTP connection request from a client to a proxy cache over a packet-based computer network, said computer program instructions comprising instructions defining the steps of:

receiving at least one packet containing the request message, the at least one packet having an IP header comprising a destination address of an origin server, the request message including a complete address of a specified object at the origin server;

modifying the request message by combining the destination address of the origin server with the complete address; and

forwarding the at least one packet containing the modified request message to the proxy cache over the packet-based network.

39. The computer readable memory of claim 37 wherein the step of modifying comprises the step of prefixing the complete address in the request message with the destination address of the origin server.

40. The computer readable memory of claim 39 wherein said computer program instructions further comprise instructions defining the step of:

translating the destination address in an IP header of the at least one packet containing the modified request message to the address of the proxy cache.

41. The computer readable memory of claim 40 wherein said computer program instructions further comprise instructions defining the step of:

translating a source address in the IP header of the at least one packet containing the modified request message from an address of the client to an address of the Layer 4 switch.

42. The computer readable memory of claim 40 wherein said computer program instructions further comprise instructions defining the step of:

in a TCP header in packets received from the proxy cache commencing with an acknowledgment to the modified request message, modifying a value in a acknowledged byte sequence number field to reflect the increased number of bytes in the modified request message resulting from the step of prefixing the complete address with the destination address of the origin server.

43. The computer readable memory of claim 42 wherein the step of modifying the value in the acknowledged byte sequence number field comprises the step of decreasing the value in the acknowledged byte sequence number field by the number of bytes added by prefixing the destination address of the origin server to the complete address in the request message.

44. The computer readable memory of claim 43 wherein said computer program instructions further comprise instructions defining the step of:

translating in the IP header the destination address to the address of the client and the source address to that of the origin server in the packets received from the proxy cache commencing with the acknowledgment to the request message.

45. The computer readable memory of claim 40 wherein said computer program instructions further comprise instructions defining the step of:

modifying a value in a sent byte sequence number field to reflect the increased number of bytes in the modified request message resulting from prefixing the complete address with the destination address of the origin server in a TCP header in packets received from the client following receipt of the request message.

46. The computer readable memory of claim 45 wherein the step of modifying the value in the sent byte sequence number field comprises the step of increasing the value in the sent byte sequence number field by the

number of bytes added by prefixing the destination address of the origin server to the complete address in the fequest message.

47. The computer readable memory of claim 46 wherein said computer program instructions further comprise instructions defining the step of:

translating the destination address to the address of the proxy cache in the IP header each packet received from the client following receipt of the request message.

48. The computer readable memory of claim of claim 38 wherein said computer program instructions further comprise instructions defining the step of selecting the proxy cache from a plurality of different proxy caches prior to receiving the at least one packet containing the request message.

49. The computer readable memory of claim 39 wherein said computer program instructions further comprises instructions defining, prior to the step of receiving the at least one packet containing the request message, the steps of:

receiving a packet from the proxy cache indicating a maximum segment size that packets sent to it thereafter should have;

reducing by a predetermined number the maximum segment size of packets to be thereafter sent to the proxy cache; and

sending a packet to the client indicating the reduced maximum segment size that packets thereafter sent by the client to the origin server should have.

50. The computer readable memory of claim 49 wherein the predetermined number is equal at least to the maximum number of bytes added by the step of prefixing the destination address to the complete address in the request message.

51. The computer readable memory of claim 50 wherein said computer program instructions further comprise instructions defining the steps of:

successively shifting to a next packet in the request message overflow bytes caused by the step of prefixing the destination address to the complete address in the request message, if the request message is contained within more than one packet; and

changing a length-of-packet parameter in the IP header in a last packet of a multi-packet request message to reflect the bytes shifted into the last packet from a previous packet.

52. The computer readable memory of claim 50 wherein said computer program instructions further comprise instructions defining the step of changing a length-of-packet parameter in the IP header to reflect the change in the length of that packet caused by the step of prefixing the destination address to the complete address in the request message, if the request message is contained within one packet.

53. The computer readable memory of claim 38 wherein the request message is a GET request.

54. The computer readable memory of claim 38 wherein the request message is a POST request.

55. The computer readable memory of claim 38 wherein the request message is a HEAD request.

FIG. 1

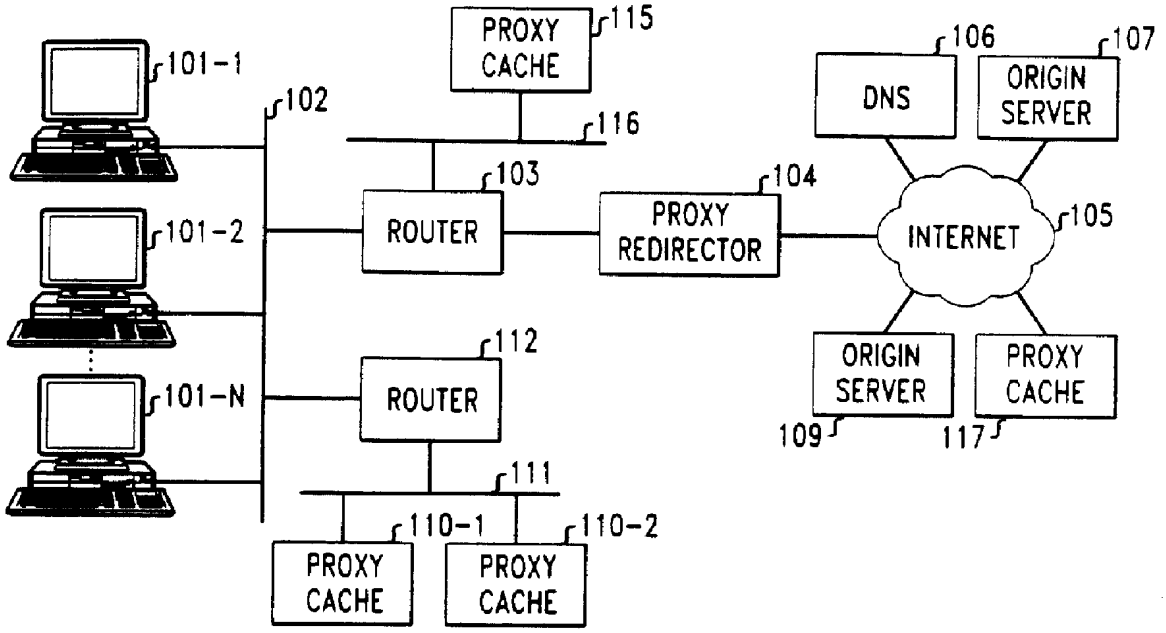

PROXY CACHE
115
116
101-1
102
101-2
101-N
ROUTER
103
PROXY REDIRECTOR
104
DNS
106
ORIGIN SERVER
107
INTERNET
105
ROUTER
112
ORIGIN SERVER
109
PROXY CACHE
117
111
110-1
110-2
PROXY CACHE
PROXY CACHE

FIG. 3

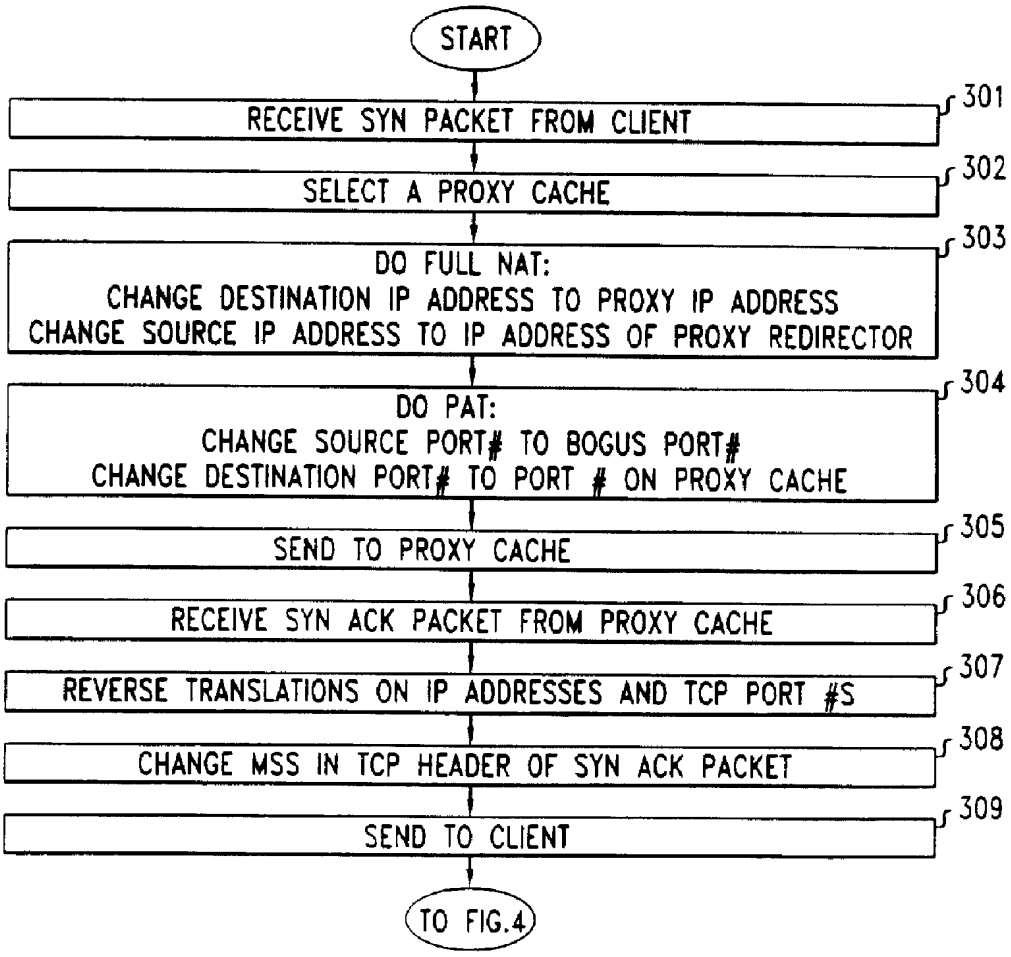

START
RECEIVE SYN PACKET FROM CLIENT
301
SELECT A PROXY CACHE
302
DO FULL NAT:
CHANGE DESTINATION IP ADDRESS TO PROXY IP ADDRESS
CHANGE SOURCE IP ADDRESS TO IP ADDRESS OF PROXY REDIRECTOR
303
DO PAT:
CHANGE SOURCE PORT# TO BOGUS PORT#
CHANGE DESTINATION PORT# TO PORT # ON PROXY CACHE
304
SEND TO PROXY CACHE
305
RECEIVE SYN ACK PACKET FROM PROXY CACHE
306
REVERSE TRANSLATIONS ON IP ADDRESSES AND TCP PORT #S
307
CHANGE MSS IN TCP HEADER OF SYN ACK PACKET
308
SEND TO CLIENT
309
TO FIG.4

FIG. 2

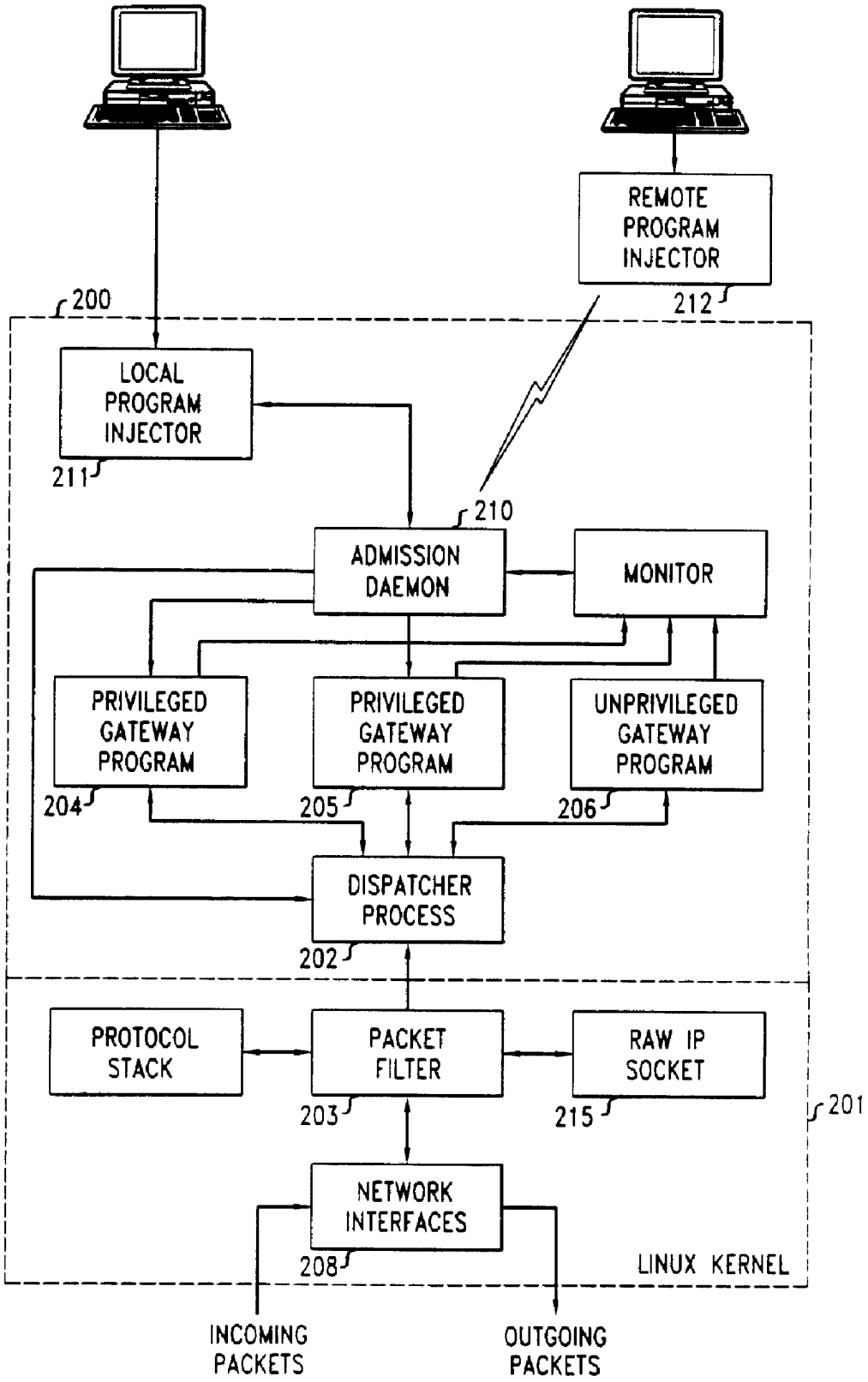

REMOTE PROGRAM INJECTOR
212
200
LOCAL PROGRAM INJECTOR
211
210
ADMISSION DAEMON
MONITOR
PRIVILEGED GATEWAY PROGRAM
204
PRIVILEGED GATEWAY PROGRAM
205
UNPRIVILEGED GATEWAY PROGRAM
206
DISPATCHER PROCESS
202
PROTOCOL STACK
PACKET FILTER
203
RAW IP SOCKET
215
201
NETWORK INTERFACES
208
LINUX KERNEL
INCOMING PACKETS
OUTGOING PACKETS

FIG. 4

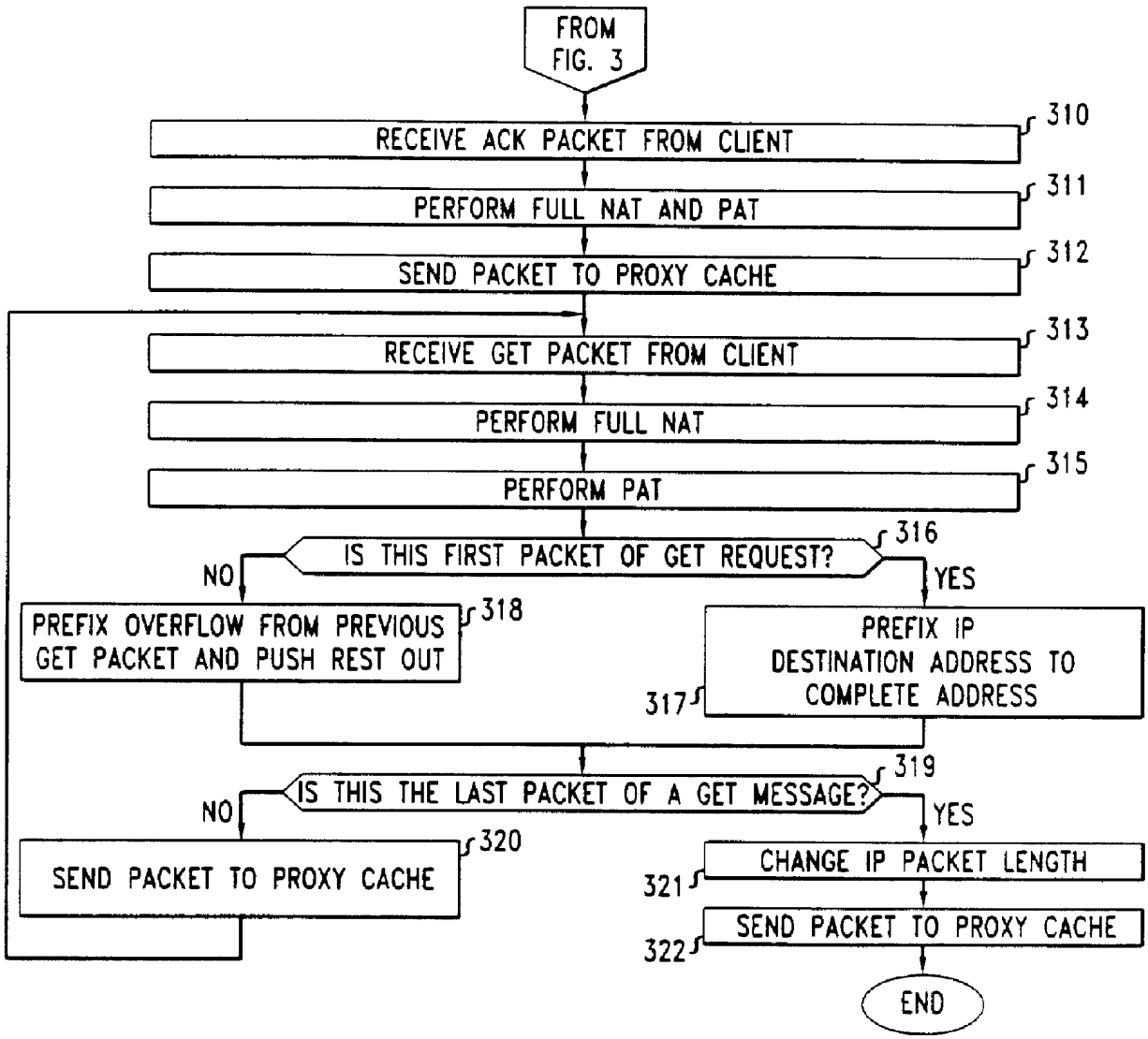

FROM FIG. 3
RECEIVE ACK PACKET FROM CLIENT
310
PERFORM FULL NAT AND PAT
311
SEND PACKET TO PROXY CACHE
312
RECEIVE GET PACKET FROM CLIENT
313
PERFORM FULL NAT
314
PERFORM PAT
315
IS THIS FIRST PACKET OF GET REQUEST?
316
NO
YES
PREFIX OVERFLOW FROM PREVIOUS GET PACKET AND PUSH REST OUT
318
PREFIX IP DESTINATION ADDRESS TO COMPLETE ADDRESS
317
IS THIS THE LAST PACKET OF A GET MESSAGE?
319
NO
YES
SEND PACKET TO PROXY CACHE
320
CHANGE IP PACKET LENGTH
321
SEND PACKET TO PROXY CACHE
322
END

FIG. 5

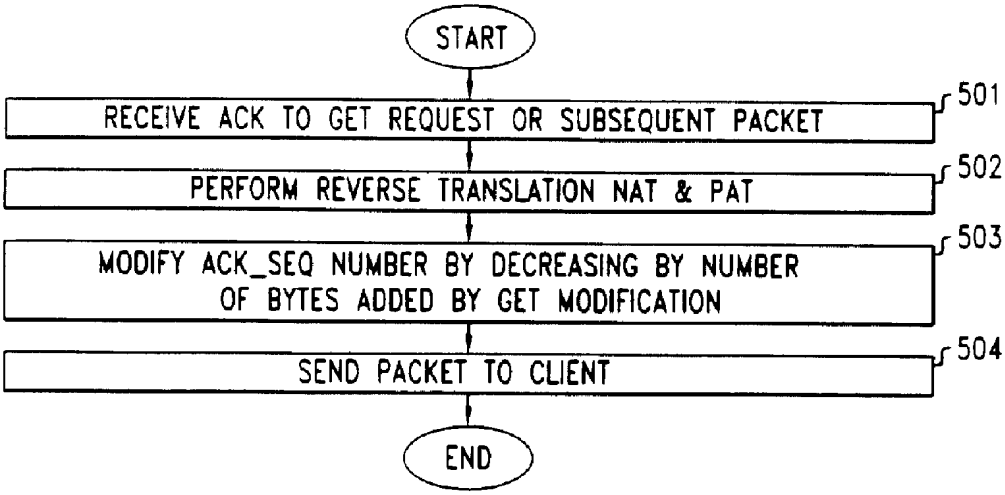
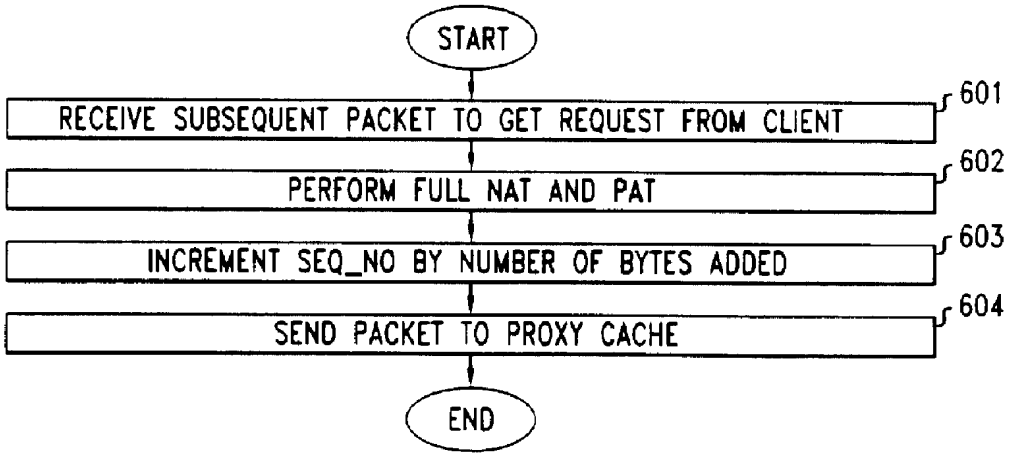

START
RECEIVE ACK TO GET REQUEST OR SUBSEQUENT PACKET
501
PERFORM REVERSE TRANSLATION NAT & PAT
502
MODIFY ACK_SEQ NUMBER BY DECREASING BY NUMBER OF BYTES ADDED BY GET MODIFICATION
503
SEND PACKET TO CLIENT
504
END

FIG. 6


START
RECEIVE SUBSEQUENT PACKET TO GET REQUEST FROM CLIENT
601
PERFORM FULL NAT AND PAT
602
INCREMENT SEQ_NO BY NUMBER OF BYTES ADDED
603
SEND PACKET TO PROXY CACHE
604
END

(19) Canadian Intellectual Property Office
An Agency of Industry Canada

Office de la Propri,t, Intellectuelle du Canada
Un organisme d'Industrie Canada

(11) CA 2 335 661 (13) A1

(40) 03.09.2001
(43) 03.09.2001

(12)

(21) 2 335 661

(51) Int. Cl.[7]: H04L 29/10, H04L 12/24

(22) 09.02.2001

(30) 09/518,356 US 03.03.2000

(71) INTERNATIONAL BUSINESS MACHINES CORPORATION,
New Orchard Road, ARMONK, XX (US).

(72) TAYLOR, LUCY ANITA (US).
ROHWER, CARY LANE (US).
BENDER, TODD ROBERT (US).
VELIAH, SUNDAR (US).
BELKNAP, WILLIAM RUSSELL (US).
FARRINGTON, SHANNON MATTHEW (US).

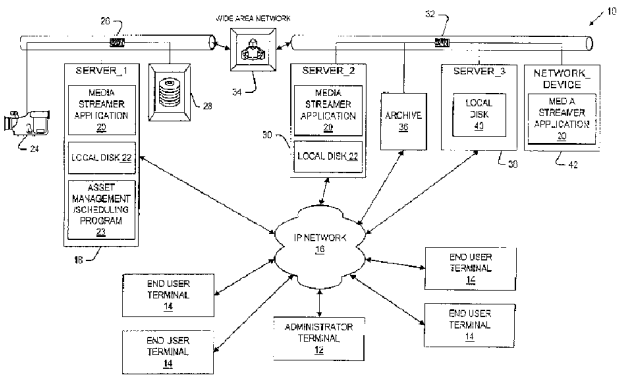
(74) BARRETT, B.P.

(54) INTERFACE UTILISATEUR GRAPHIQUE SERVANT A LA GESTION DES RESSOURCES ET A L'ORDONNANCEMENT D'UN SYSTEME DE DIFFUSION EN CONTINU

(54) ASSET MANAGEMENT AND SCHEDULING GRAPHICAL USER INTERFACE FOR MEDIA STREAMER

(57)

A process of managing media data in a network is provided, the system including an administrator terminal, at least one media server communicatively coupled to the administrator terminal via a network, and a plurality of end user terminals, each of the media servers being operative to store media data, and to transfer portions of media data to selected locations via the network, at least one of the media servers also being operative to stream media data to selected ones of the end user terminals. The process provides a centralized user interface for scheduling and coordinating the execution of operations performed by the media servers. The process begins with a step of displaying a graphical user interface including a plurality of interface components on a display unit of the administrator terminal, the components enabling a user of the administrator terminal to define and schedule operations to be performed by selected ones of the media servers on corresponding portions of media data. User input includes: server identification information indicative of a selected media server; operation information specifying an operation to be performed by the selected media server on a selected portion of media data, and schedule information specifying a schedule for performing the specified operation. Commands and associated parameters are generated based on the operation information and schedule information, the commands and associated parameters for instructing the selected media server to execute the specified operation in accordance with the specified schedule.



Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2335661 A1 2001/09/03

(21) 2 335 661

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(22) Date de dépôt/Filing Date: 2001/02/09

(41) Mise à la disp. pub./Open to Public Insp.: 2001/09/03

(30) Priorité/Priority: 2000/03/03 (09/518,356) US

(51) Cl.Int.^{7}/Int.Cl.^{7} H04L 29/10, H04L 12/24

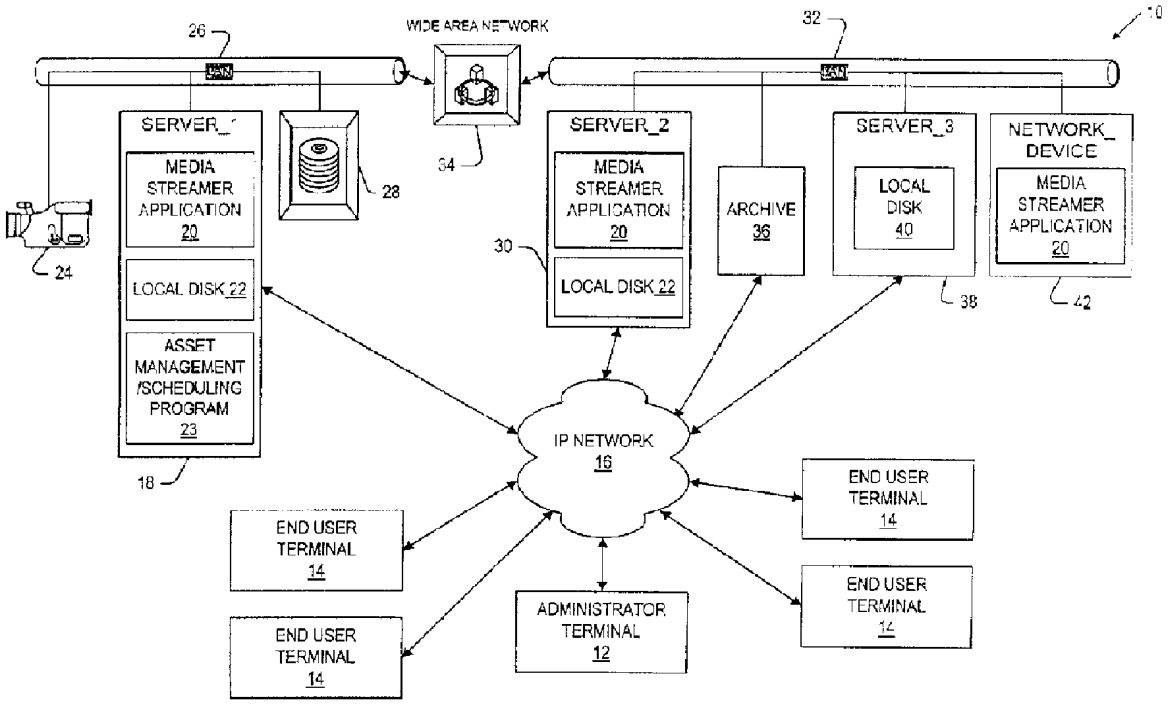
(71) Demandeur/Applicant:
INTERNATIONAL BUSINESS MACHINES CORPORATION, US

(72) Inventeurs/Inventors:
VELIAH, SUNDAR, US;
TAYLOR, LUCY ANITA, US;
ROHWER, CARY LANE, US;
FARRINGTON, SHANNON MATTHEW, US;
BENDER, TODD ROBERT, US;
BELKNAP, WILLIAM RUSSELL, US

(74) Agent: BARRETT, B.P.

(54) Titre : INTERFACE UTILISATEUR GRAPHIQUE SERVANT A LA GESTION DES RESSOURCES ET A L'ORDONNANCEMENT D'UN SYSTEME DE DIFFUSION EN CONTINU

(54) Title: ASSET MANAGEMENT AND SCHEDULING GRAPHICAL USER INTERFACE FOR MEDIA STREAMER



(57) Abrégé/Abstract:
A process of managing media data in a network is provided, the system including an administrator terminal, at least one media server communicatively coupled to the administrator terminal via a network, and a plurality of end user terminals, each of the media servers being operative to store media data, and to transfer portions of media data to selected locations via the network, at least one of the media servers also being operative to stream media data to selected ones of the end user terminals. The process provides a centralized user interface for scheduling and coordinating the execution of operations performed by the media servers. The process begins with a step of displaying a graphical user interface including a plurality of interface

(21) 2 335 661

(13) A1

(57) Abrégé(suite)/Abstract(continued):
components on a display unit of the administrator terminal, the components enabling a user of the administrator terminal to define and schedule operations to be performed by selected ones of the media servers on corresponding portions of media data. User input includes: server identification information indicative of a selected media server; operation information specifying an operation to be performed by the selected media server on a selected portion of media data, and schedule information specifying a schedule for performing the specified operation. Commands and associated parameters are generated based on the operation information and schedule information, the commands and associated parameters for instructing the selected media server to execute the specified operation in accordance with the specified schedule.

ASSET MANAGEMENT AND SCHEDULING GRAPHICAL USER INTERFACE FOR MEDIA STREAMER

ABSTRACT

A process of managing media data in a network is provided, the system including an administrator terminal, at least one media server communicatively coupled to the administrator terminal via a network, and a plurality of end user terminals, each of the media servers being operative to store media data, and to transfer portions of media data to selected locations via the network, at least one of the media servers also being operative to stream media data to selected ones of the end user terminals. The process provides a centralized user interface for scheduling and coordinating the execution of operations performed by the media servers. The process begins with a step of displaying a graphical user interface including a plurality of interface components on a display unit of the administrator terminal, the components enabling a user of the administrator terminal to define and schedule operations to be performed by selected ones of the media servers on corresponding portions of media data. User input includes: server identification information indicative of a selected media server; operation information specifying an operation to be performed by the selected media server on a selected portion of media data, and schedule information specifying a schedule for performing the specified operation. Commands and associated parameters are generated based on the operation information and schedule information, the commands and associated parameters for instructing the selected media server to execute the specified operation in accordance with the specified schedule.

STL9-1999-0085

ASSET MANAGEMENT AND SCHEDULING GRAPHICAL USER INTERFACE FOR MEDIA STREAMER

BACKGROUND OF THE INVENTION

Field of the Invention:

The present invention relates generally to a method and apparatus for managing media data in computer network systems. More specifically, the present invention relates to a process and apparatus providing a centralized graphical user interface for managing media assets in a network by specifying and scheduling operations to be performed by a plurality of media servers in a computer network.

Description of the Prior Art:

A media server is a network server system which is operative to perform various operations on media data files in a computer network system. Digital media files may include video data, audio data, text data, and graphical data. Typically, a media server is operative to perform media operations including: delete operations for deleting media assets from locations in a network; copy operations for copying media assets from source locations to destination locations in a network; multicasting operations for streaming media assets from the media server to clients via the network; and encoding operations for encoding media data, received by the media server from multimedia equipment such as a video camera, in order to create media assets.

For purposes of the present application, a media asset is defined to include parsed media data which is configured to be streamed from a media server to one or more clients in a particular type of network. One example of a media server is IBM's VideoCharger™ server and media streamer which combines functions of digital computing and the video broadcast industry into a versatile, cost-effective system for high-quality storage and delivery of multimedia content. Details of the VideoCharger™ product are disclosed in U.S. patent application Ser. No.

08/961,704, entitled "Scalcable Network Transparent Access Framework for Multimedia Serving," filed on or about Oct. 31, 1997, and U.S. patent application Ser. No. 08/961,706, entitled "Seamless Playlist," filed on or about Oct. 31, 1997, each of which is hereby incorporated.

The VideoCharger™ server, provided by International Business Machines Corp., provides one example of a media server. The VideoCharger™ server provides for the delivery of continuous time media data (i.e. audio and/or video data) to Internet or Intranet connected clients. The video is "streamed", (i.e. delivered in real-time) and does not require that the file be downloaded or saved before being played by the client software. In video parlance, the video is "pushed" by the server over the network to the client. This approach is different from most file servers where the data is "pulled" by the client issuing successive "reads" to the server. The "push" architecture is similar to a broadcast environment where a video stream is started by a play command and will continue until stopped. U.S. Patent 5,918,020, issued to Blackard et al. on June 29, 1999 describes a data processing system and a method for implementing a push, or streaming, model for communicating time sensitive encoded data, such as video and audio data, in a communication network. In addition to supporting industry standard file formats for audio and video, the VideoCharger media server supports the popular Internet and World Wide Web (WWW) protocols including IP and Hyper Text Transport Protocol (HTTP). This allows the product to be used with Industry standard applications like HTML Web Browsers. It also allows the product to be used on a wide variety of network types including LANs (Local Area Networks such as Ethernet, Token Ring, FDDI), WANs (Wide Area Networks such as T1, E1, T3, E3) and ATM (Asynchronous Transfer Mode).

An IP multicast feature of the VideoCharger™ media server allows it to be used as a broadcast type server in the Internet environment. This allows a single audio or video stream to be sent to multiple clients, thereby reducing the bandwidth requirements on the network. In addition, VideoCharger on Windows NT offers embedded encoder support. This allows an

MPEG encoder to be installed on the server and have VideoCharger directly control the encoder for functions such as real-time IP Multicast and real-time IP Multicast with live recording of the same stream. This is a very efficient yet powerful method of providing a broadcast of a live event while recording it for later re-broadcast with minimal network load. IBM also provides a VideoCharger Server for AIX.

The functions and capabilities of media servers are very useful to archivists, film/video production groups, educational institutions, research groups, medical technologists, advertising and creative agencies, print and Web publishers, and marketing communications agencies. In accordance with prior art asset management and scheduling methods, an administrator of a media server having access rights thereto may configure the media server to execute specified media operations to be performed by that server in accordance with a specified schedule.

In accordance with one prior art asset management and scheduling method, a administrator may configure a particular media server to perform particular types of media operations by downloading hypertext markup language (HTML) Web pages via corresponding uniform resource locators (URL's) from the particular media server, each of the HTML Web pages providing an interface for scheduling a corresponding one of the particular types of media operations to be executed by the particular media server. A copy/delete HTML Web page provides rudimentary interface functions allowing the administrator to define and schedule copy operations and delete operations to be performed by the particular media server. The copy/delete HTML Web page includes: prompts and active areas enabling the user to specify a copy operation or a delete operation; and text boxes enabling the administrator to enter a path to a selected media file which the subject of the copy or delete operation. An encoding HTML Web page provides rudimentary interface functions allowing the administrator to activate an encoding operation to be performed by the particular media server. The encoding HTML Web page includes text boxes enabling the administrator to enter: a path and a selected file name for the asset to be encoded; and a duration value indicating a duration for the encoding operation. A

problem associated with this prior art method of remotely activating an encoding operation is that the administrator is not provided with any view of the scene to be encoded. A multicasting HTML Web page provides rudimentary interface functions allowing the administrator to schedule a multicasting operation to be performed by the particular media server. The multicasting HTML Web page includes text boxes enabling the administrator to enter: a path indicating a selected asset to be multicasted; one internet protocol (IP) address and port number for multicasting the selected asset; a scheduled start date and start time for executing the multicasting operation; and a duration value indicating a time duration for the multicasting operation.

One problem associated with the above described prior method of configuring media servers to manage media assets and schedule media operations is that there are no mechanisms which prevent the administrator from specifying invalid options, such as specifying an invalid path or an invalid file name for a selected asset. For example, in the course of defining a copy operation using the copy HTML Web page, if the administrator specifies an incorrect path for an asset to be copied, then the copy operation will not be successfully executed at the scheduled time. Also, the prior art interface consisting of the HTML Web pages described above provides no means for notifying the administrator as to whether or not the operation is successfully completed upon passing of the scheduled time for executing the operation. Furthermore, a failure of a particular copy operation would result in a failure of a subsequently scheduled media operation which is dependent upon the particular copy operation, as further explained below.

For a variety of multimedia network applications, it is important to be able to coordinate the scheduling of media operations to be performed by a plurality of media servers. As an example, an administrator of an educational institution such as a college or university may desire to configure one or more media servers to create, store, manage, and stream media assets. In this example, the media assets may include video data and audio data comprising a lecture given by a professor in a class room. The administrator may configure a first server, that is an encoding

server, to encode media data received from multimedia equipment, such as a video camera, at a scheduled time. Alternatively, the administrator may manually control the encoding media server using a local programming interface residing at the server site to start, stop, and pause encoding of the media data generated by the camera while viewing a local screen which displays the lecture being recorded. However, there is no means provided in the prior art for remotely controlling the encoding of media assets by a media server. After the administrator encodes a new asset at an encoding media server, or schedules an encoding operation to be performed at the encoding server, the administrator may configure the encoding server to execute further operations including: storing the encoded asset in a storage device; and multicasting the encoded asset to a selected group of clients either in real time or at a later time in accordance with a programmed schedule. Further, the administrator may configure the encoding server to transfer the encoded asset to a second media server via a network.

In the above described prior art asset management and scheduling method, in order to coordinate the scheduling of media operations at the first and second servers, the administrator must access and configure each server separately. As the number of scheduled operations increases, and the number of configured media servers increases, it becomes more and more difficult for the administrator to coordinate the scheduling of the operations between the multiple servers because there is no centralized location provided in the prior art for automatically logging those operations which are specified and scheduled by the administrator to be executed at the multiple servers. A log of operations scheduled to be executed by each particular media server is stored in a log file maintained by the operating system of the particular media server (e.g., a Windows NT log file, or an AIX log file). However, the administrator must access the log file of each particular server separately in order to review operations to be performed by the particular media server.

In one particular scenario, an administrator may wish to configure a first server to: encode media data to create a new asset; and then copy the new asset at a later scheduled time to

a second server. Provided that the administrator has access to the second server, the administrator may: configure the second server to multicast the encoded asset to a selected group of clients in accordance with a programmed schedule; and configure the second server to subsequently remove the encoded asset from the catalog of the second server so that the new asset cannot be accessed after a certain period of time. The administrator may wish to configure the two servers to perform the steps of creating a new asset at the first media server, transferring the new asset to the second server, and multicasting it from the second server in order to achieve the goal of minimizing network congestion problems which would arise in streaming directly from the first encoding server to the selected clients. Note that the success of the multicasting operation scheduled for execution by the second server is dependent upon the success of the copy operation scheduled for execution by the first server. As mentioned above, a failure of the copy operation would result in a failure of the subsequently scheduled multicasting operation which is dependent upon the copy operation. Because there is no means for notifying the administrator as to whether or not the copy operation is successfully completed, the administrator also would have no warning that the dependent multicasting operation would inevitably fail as well.

In the above described prior art asset management and scheduling method, in order to coordinate the scheduling of media operations at the first and second servers, the administrator could separately access the log files of the first and second servers, and review each. However, this process is very cumbersome. If the administrator needs to coordinate and schedule events at a multiplicity of media servers, which may be residing in different time zones, the coordination and management of the scheduled events at the multiplicity of servers becomes very difficult.

Therefore, one problem with the prior art asset management and scheduling processes is that an administrator of a plurality of media servers is required to interface separately with each media server in order to read the log files for the purposes of coordinating scheduled operations at each of the media servers. Another problem is that the administrator is prone to specify invalid paths for assets to be operated upon thereby resulting failure of media operations. A

further problem is that there is no notification system for notifying the administrator in the event of success or failure of the scheduled operations.

SUMMARY OF THE INVENTION

It is an object of the present invention to provide a graphical user interface process providing a centralized interface for remotely managing media assets, and scheduling media operations to be performed by a plurality of media servers in a computer network system, the media operations including deleting media assets from a source location in a network, copying media assets from selected source locations to selected destination locations; multicasting operations for streaming media assets from selected media servers to selected clients via the network; and encoding operations for encoding media assets.

Briefly, a presently preferred embodiment of the present invention provides a process of managing media data in a network system including an administrator terminal, at least one media server communicatively coupled to the administrator terminal via a computer network, and a plurality of end user terminals communicatively coupled to the administrator terminal and to the media servers via the network, each of the media servers being operative to access a memory device for storing media data, and to transfer portions of media data to selected locations via the network, at least one of the media servers being operative to stream media data to selected ones of the end user terminals. At least one of the media servers is also operative to encode and parse portions of media data to create media assets. Also, each of the media servers is further operative to maintain a catalog including a plurality of pointers to catalogued ones of the media assets stored in a local disk included within the media server.

The process provides a centralized user interface for scheduling and coordinating the execution of operations performed by selected ones of the media servers, each of the operations being associated with a corresponding portion of media data. The process begins with a step of displaying a graphical user interface including a plurality of interface components on a display

unit of the administrator terminal, the components enabling a user of the administrator terminal to define and schedule media operations to be performed by selected ones of the media servers on corresponding portions of media data. User input is received via an input/output unit of the administrator terminal. The user input includes: server identification information indicative of a selected media server; operation information specifying an operation to be performed by the selected media server on a selected portion of media data, and schedule information specifying a schedule for performing the specified operation. The process further includes the steps of: generating commands and associated parameters based on the operation information and the schedule information, the commands and associated parameters for instructing the selected media server to execute the specified operation in accordance with the specified schedule; and transmitting the commands and the associated parameters to the selected media server via the network.

In an embodiment, the administrator terminal also includes a processing unit, and a browser application executed by the processing unit. The process is initiated by performing the steps of: transmitting an applet to the administrator terminal via the network; and executing the applet over the processing unit of the administrator terminal; whereby the graphical user interface is displayed within a browser window generated by the browser application on the display unit.

The operations include: delete operations for deleting portions of media data from catalogs of predetermined mapped ones of the media servers; copy operations for copying portions of media data from selected source ones of the locations to selected destination ones of the locations; multicasting operations for streaming portions of media data from selected ones of the media servers to selected ones of the end user terminals via the network; and encoding operations for encoding media data received by selected ones of the media servers.

The process includes a feature for displaying graphical indicia indicative of a plurality of user scheduled operations that have been previously defined and scheduled by the user, each of

the scheduled operations being associated with corresponding previously specified information including previously specified server identification information, previously specified operation information, and previously specified schedule information. The process further comprises the steps of: providing interface components allowing the user to select one of the displayed scheduled operations; receiving user input indicative of a selected scheduled operation; determining a portion of the interface components of the graphical user interface that is associated with the selected scheduled operation; and displaying the determined portion of the interface components having previously specified information associated with the selected scheduled operation displayed therein, the displayed portion of the interface components enabling the user to edit the previously specified information associated with the selected scheduled operation in order to generate revised information associated with a revised operation, the revised information including revised server identification information, revised operation information, and revised schedule information.

The foregoing and other objects, features, and advantages of the present invention will be apparent from the following detailed description of the preferred embodiment which makes reference to the several figures of the drawing.

IN THE DRAWINGS:

FIG. 1 is a block diagram illustrating a networked computer system including a plurality of network locations for storing and operating on media data, and an administrator terminal communicatively coupled with the locations via a network, the administrator terminal providing an asset management and scheduling graphical user interface (GUI) process in accordance with the present invention for remotely managing operations including generating, copying, and streaming media data at and between selected ones of the locations;

FIG. 2 is a block diagram illustrating an exemplary computer system used to implement the administrator terminal of FIG. 1;

FIG. 3 is a block diagram illustrating a main GUI screen of the asset management and scheduling GUI process, the main GUI screen providing for a user to select from management and scheduling options including managing assets, encoding assets, and multicasting existing assets;

FIG. 4 is a flow diagram illustrating a sub-process for implementing the main GUI screen of FIG. 3;

FIG. 5 is a block diagram illustrating an asset management GUI screen providing for the user to select from asset management options including mapping new locations, deleting available locations, and copying and deleting selected assets;

FIG. 6 is a flow diagram illustrating a sub-process of the GUI process of the present invention for managing assets using the asset management GUI screen of FIG. 5;

FIGS. 7A through 7E are a block diagrams illustrating location mapping GUI screens providing for the user to map locations including local and remote storage units, media servers, and archives;

FIGS. 8A through 8D are flow diagrams illustrating sub-processes for mapping locations using the GUI screens of FIGS. 7A through 7E;

FIG. 9 is a flow diagram illustrating a sub-process for deleting selected locations from an available locations list;

FIGS. 10A through 10D are flow diagrams illustrating sub-processes for copying and deleting selected assets located at selected locations using the asset management GUI screen of FIG. 5;

FIG. 11 is a block diagram illustrating a rename GUI screen for renaming a selected asset before copying the selected asset;

FIG. 12 is a block diagram illustrating an encoding GUI screen for encoding a new media asset;

FIGS. 13A and 13B are flow diagrams illustrating a sub-process of encoding and optionally recording media data in accordance with the present invention using the GUI screen of FIG. 12;

FIG. 14 is a block diagram illustrating a GUI screen in accordance with the present invention for defining encoder attributes;

FIG. 15 is a flow diagram illustrating a sub-process of defining encoder attributes using the GUI screen of FIG. 14;

FIG. 16 is a block diagram illustrating a GUI screen for manually and remotely controlling the encoding of a new asset;

FIG. 17 is a flow diagram illustrating a sub-process of manually controlling the encoding of an asset from a remote location in accordance with the present invention;

FIG. 18 is a flow diagram illustrating sub-process of defining and scheduling streaming operations in accordance with the present invention;

FIG. 19 is a block diagram illustrating a GUI screen for defining a destination group of IP addresses and port numbers for streaming operations;

FIG. 20 is a flow diagram illustrating a sub-process of defining the destination group of IP addresses and port numbers using the GUI screen of FIG. 19;

FIG. 21 is a flow diagram illustrating a sub-process of specifying and scheduling a recording operation for recording an asset;

FIG. 22 is a block diagram illustrating a playback scheduling GUI screen;

FIGS. 23A through 23D are flow diagrams illustrating a play-back sub-process for defining and scheduling a streaming operation for streaming a new asset using the scheduling GUI screen of FIG. 22;

FIG. 24 is a block diagram illustrating a GUI screen for defining an interval for a playback schedule;

FIG. 25 is a flow diagram illustrating a sub-process for defining a playback schedule interval using the screen of FIG. 24;

FIG. 26 is a block diagram illustrating a GUI screen for defining notification e-mail messages to be sent to selected end users upon detection of a selected event associated with streaming operations;

FIG. 27 is a flow diagram illustrating a process of specifying events for which notification e-mail messages are to be sent to selected end users;

FIG. 28 is a block diagram illustrating a GUI screen for defining groups of destination e-mail addresses for the purpose of sending notification messages to the destination addresses;

FIG. 29 is a flow diagram illustrating a sub-process of defining a group of destination e-mail addresses;

FIG. 30 is a block diagram illustrating an edit message GUI screen;

FIG. 31 is a flow diagram illustrating a process of editing a message using the screen of FIG. 30;

FIG. 32 is a block diagram illustrating a GUI screen providing an interface for defining and scheduling multicasting operations;

FIGS. 33A through 33E are flow diagrams illustrating a sub-process of defining and scheduling streaming operations in accordance with the present invention using the GUI screen of FIG. 32;

FIG. 34 is a block diagram illustrating a GUI screen providing an interface for selecting a date;

FIG. 35 is a block diagram illustrating a global schedule summary Web document in accordance with the present invention;

FIG. 36 is a flow diagram illustrating a process for managing global schedule information in accordance with the present invention;

FIG. 37 is a flow diagram illustrating a process for generating the global schedule Web document of FIG. 35; and

FIG. 38 is a flow diagram illustrating a process for reviewing a summary of all scheduled media operations associated with a particular user of the asset management and scheduling GUI process of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

FIG. 1 shows a block diagram illustrating an exemplary networked computer system at 10 including an administrator terminal 12 configured in accordance with the present invention for implementing an asset management and scheduling GUI process for remotely managing and scheduling media operations including generating, copying, and multicasting media assets at and between a plurality of locations in the system 10 as further explained below.

The depicted system 10 also includes: a plurality of end user terminals 14 communicatively coupled with the administrator terminal 12 via an Internet protocol (IP) network 16 such as the public Internet or a private internet; a first media server 18 designated SERVER_1 communicatively coupled with the administrator terminal 12 via the IP network, the first media server having a media streamer application 20 which is executed by the server, an asset management and scheduling program 23 in accordance with the present invention which is executed by the server, and a local disk 22 providing memory storage for media data including audio data, video data, and text; a video camera 24 for generating media data having video data and audio data, the camera 24 being communicatively coupled with the first media server 18 via a first local area network (LAN) 26; a remote disk 28 communicatively coupled with the first media server 18 via the first LAN 26; a second media server 30 designated SERVER_2 communicatively coupled with the administrator terminal 12 via the IP network, the second server having a media streamer application 20 which is executed by the server, and a local disk

22 for storing media data; an archive data storage unit 36 communicatively coupled with SERVER_2 via a second LAN 32 which is communicatively coupled with the first LAN 26 via a wide area network (WAN) 34; a server 38 designated SERVER_3 communicatively coupled with the administrator terminal 12 via the IP network, and also being coupled with the second LAN 32, the third server having a local disk 40 for storing media data; and a network device 42 having a media streamer application 20 which is executed by the network device 42, the device 42 being communicatively coupled with SERVER_2 via the second LAN 32.

Each of the media streamer applications 20 is operative to encode and parse portions of media data to create a media asset, and is also operative to stream assets to selected groups of the plurality of end user terminals 14. The parsing of the media data includes adding headers and attributes to packets of the media data in order to allow for the detection of failures in staging, that is copying, of media assets from one media server to another media server via the IP network 16 or via the WAN 34. Each of the media streamer applications 20 is executed by a corresponding processing unit (not shown) of the corresponding one of the media servers 18 and 30, and the network device 42.

As further explained below, the end user terminals 14 include plug-in applications which, when executed along with a viewing application such as a browser, are operative to decode and play the assets in real time.

In accordance with the present invention, an asset management and scheduling graphical user interface (GUI) process is provided at the administrator terminal 12 as described in detail below. In general, the asset management and scheduling GUI process allows for a user of the administrator terminal 12 to remotely define and schedule the execution of a variety of operations on media data at selected locations (e.g., selected servers, archives, and networked devices) in the networked computer system 10.

In accordance with the asset management and scheduling GUI process, interface components are displayed on a display unit of the administrator terminal 12, the components

prompting a user of the administrator terminal to define and schedule operations to be performed by selected ones of the media servers on corresponding selected assets. In response to the interface components, user input is received via an input/output of the administrator terminal 12. The user input generally includes: operation information specifying details of a selected event, or operation, to be performed by a selected one of the media servers 18 and 30 on a selected asset which is accessible by the selected media server; and schedule information associated with the operation information, the schedule information indicating an associated schedule for performing the selected operation. In accordance with the asset management and scheduling GUI process, commands and associated parameters are generated based on the operation information and the associated schedule information. The commands and associated parameters are transmitted to the corresponding selected one of the media servers 18 and 30 for execution of the specified operation in accordance with the associated schedule.

Operations which may be remotely managed and scheduled by the asset management and scheduling GUI include: remotely controlling encoding operations for encoding media assets at selected locations in the system 10 by manual remote control, or by scheduling one or more time delayed encoding operations; streaming assets from a selected source media streamer to a selected group of destination end users; "copying" media data from a selected source location in the system 10 setting to a selected destination location; and "deleting" media assets from selected locations.

The media streamer applications 20 are operative to execute several different types of copy actions for copying media assets including: "adding" media files or assets from a local disk 22 to a "catalog" of a corresponding media streamer application 20, the catalog being a list of files and/or assets which the media streamer application may access; "loading" media files or assets from the remote disk 28 to the catalog of the corresponding media streamer application 20 at SERVER_1; and "staging" media assets from a selected source one of the media servers 18 and 30 to a selected destination one of the media servers. In the preferred embodiment, in order

to allow for detection of a failure in a staging operation, only media assets which include parsed media data, as opposed to media files which include unparsed media data, are staged via the IP network 16 or via the WAN 34.

The "delete" operations include removing selected assets from the catalog of a selected media streamer application 20 at the selected media server. Media files often comprise intellectual property, and therefore, it is desirable to be able to remove asset from the catalogs of media servers so that the assets cannot be retrieved by persons who are not authorized to do so.

In a scheduled encoding mode, the asset management and scheduling GUI provides for defining and scheduling the activation of the camera 24 in accordance with a user defined schedule to generate video data which is encoded by the media streamer application 20 at SERVER_1. The user may schedule a later date and time for activating the video camera and the encoding process; specify a location to which the encoded data is to be recorded, or stored; specify that the media data is to be streamed out in real time to selected destination locations; and specify that e-mail messages are to be sent to selected end users upon success or failure of the streaming. In a manual encoding mode, the user of the asset management and scheduling GUI controls the encoding process at SERVER_1 while viewing the scene to be encoded from the administrator terminal 12.

In the preferred embodiment of the present invention, the asset management and scheduling GUI process is implemented as a Java applet executed at the administrator terminal 12, the applet being received from the asset management and scheduling program 23 at SERVER_1. In an alternative embodiment of the present invention, the asset management and scheduling GUI process is implemented as a Java application executed at the administrator terminal 12, the application being implemented by executing computer readable instructions stored at the administrator terminal 12. In yet another embodiment of the present invention, the asset management and scheduling GUI process is implemented by logic shared between the administrator terminal 12 and SERVER_1 so as to maximize efficiency.

As further explained below, each of the plurality of end user terminals 14 includes a browser application running thereon for viewing assets including media data (e.g., an MPEG video clip) streamed to the end user terminal from corresponding ones of the media servers 18 and 30. In one embodiment of the present invention, each of the end user terminals 14 has a media player stored therein, and the browser running on the end user terminal loads the media player in order to view the media data. The media player includes a decoder for decoding the media data. In another embodiment, the browsers running on the end user terminals 14 natively understand the encoding format of the asset streamed thereto and a plug-in is not required.

FIG. 2 shows a block diagram illustrating an exemplary general purpose computer system at 50 which may be used to implement each of the administrator terminal 12, the end user terminals 14, the media servers 18 and 30, and the network device 42 (FIG. 1). The computer system 50 includes: a processing unit 52 communicatively coupled to a bus 54; a random access memory (RAM) unit 56 communicatively coupled to the processing unit via the bus; a read only memory (ROM) unit 58 communicatively coupled to the processing unit via the bus; a main memory unit 60, such as a hard disk or other memory storage device, communicatively coupled to the processing unit via the bus; a display unit 62, such as a cathode ray tube or flat panel display, communicatively coupled to the bus for displaying graphical information; an input/output unit (I/O unit) 64, such as a keyboard and mouse, coupled with the processing unit via the bus, and providing an interface for a user to provide input; and a modem 66 coupled with the bus, and providing communication with remote devices via a computer network such as the IP network (FIG. 1).

In the preferred embodiment of the present invention, the main memory unit 60 (FIG. 2) of the administrator terminal 12 (FIG. 1) stores computer readable instructions for implementing a viewing application supporting a Java interpreter (e.g., Netscape Navigator, Microsoft Explorer, or Sun's HotJava browser). The Java interpreter is operative to execute compiled Java byte code, also called J-code. In the preferred embodiment of the present invention, the asset

management and scheduling GUI process is implemented as a Java applet which is provided to the administrator terminal 12 (FIG. 1) via the IP network 16 (FIG. 1) as executable content inside of a Web document.

Also, in the preferred embodiment of the present invention, the main memory unit 60 (FIG. 2) of each of the end user terminals 14 (FIG. 1) stores computer readable instructions for implementing a viewing application such as a Web browser (e.g., Netscape Navigator, Microsoft Explorer), and computer readable instructions for implementing a plug-in. Further, in the preferred embodiment of the present invention, the main memory unit 60 (FIG. 2) of each of the media servers 18 and 30 (FIG. 1) stores computer readable instructions for implementing the media streamer application 20 (FIG. 1), and instructions for implementing the asset managing and scheduling program 23 (FIG. 1).

With reference back to FIG. 1, initializing steps of the asset management and scheduling process of the present invention include sending a request for a predetermined Web page from the Web browser running on the administrator terminal 12 to the asset management and scheduling program 23 at SERVER_1 via the IP network 16. The management and scheduling program 23 packages the predetermined Web page, along with an asset management and scheduling GUI process applet in accordance with the present invention, for transmission to the Web browser running on the administrator terminal 12. The asset management and scheduling applet is operable to create and manage an embedded menu in a displayed Web page when the Web page is displayed and the applet is executed by the Web browser running on the administrator terminal 12. In the preferred embodiment of the present invention, the asset management and scheduling applet is coded in Java™ and the Web browser supports a Java interpreter.

In a preferred embodiment, each of the media streamer applications 20 provides for encoding media data and parsing the media data to create media assets including encoded and parsed media data. Also in the preferred embodiment, the media players loaded by the browsers

running on the end user terminals 14 are configured to decode and display the media assets streamed thereto by the media streamer application 20.

The asset management and scheduling GUI provided by the administrator terminal 12 is described below with reference to flow diagrams. In the preferred embodiment, the GUI process is implemented as a Java applet executed by the processing unit 52 (FIG. 2) of the administrator terminal 12 (FIG.1) to display a series of GUI screens within a Web browser window on the display unit 62 (FIG. 2) of the administrator terminal. A user of the administrator terminal 12 (FIG. 1) may interface with the below described GUI using the I/O units 64, such as a keyboard and mouse, to provide user input by activating various buttons and check boxes, and by entering and editing text as required and prompted by the GUI screens.

In the below described embodiment of the present invention, each of a plurality of interface functions provided by each of the GUI screens is implemented using a corresponding GUI component selected from a variety of GUI components including radio buttons, check boxes, drop-down lists, spin buttons, editable text boxes, non-editable text boxes, etc. However, as is well understood to one of ordinary skill in the art, each of the interface components of each of the GUI screens described below may alternatively be implemented using other types of GUI components. The asset management and scheduling GUI process of the present invention is more generally described with reference to the flow diagrams while the depicted GUI screens illustrate a currently preferred embodiment.

FIG. 3 shows a block diagram illustrating a main graphical user interface screen (GUI screen) at 100, the screen 100 being used in the asset management and scheduling GUI process of the present invention. The main GUI screen 100 includes an asset management icon 102, an "encode a new asset" icon 104, and a multicast existing assets icon 106.

FIG. 4 shows a flow diagram illustrating a sub-process at 110 for providing a graphical user interface allowing for a user to select from a plurality of main options of the asset management and scheduling GUI. In the preferred embodiment, sub-process 110 is implemented

using the main GUI screen 100 (FIG. 3). The depicted process begins with step 112 in which a user of the administrator terminal 12 (FIG. 1) is prompted to select from main options including an asset management option, an option for encoding a new asset, and an option for multicasting existing assets. In step 114, user input indicating a selected one of the main options is received in response to the user selecting one of the three icons 102, 104, and 106 of the main GUI screen (FIG. 3).

From step 114, the process proceeds to 116 at which it is determined whether the asset management icon 102 (FIG. 3) has been selected, and if so, the process proceeds to "A" (to FIG. 6) to execute an asset management and scheduling GUI sub-process as further explained below. If it is determined at 116 that the asset management option has not been selected, the sub-process proceeds to 118 at which it is determined whether the icon 104 (FIG. 3) for encoding a new asset has been selected, and if so, the depicted process proceeds to "B" (to FIG. 13A) to execute a GUI sub-process for encoding a new asset as further explained below. If is determined at 116 and 118 that neither of the icons 102 and 104 (FIG. 3) of the main GUI screen have been selected, the sub-process proceeds to 120 at which it is determined whether the icon 106 (FIG. 3) for multicasting existing assets has been selected, and if so, the depicted process proceeds to "C" (to FIG. 33A) to execute a GUI sub-process for multicasting existing assets.

FIG. 5 shows a block diagram illustrating an asset management GUI screen at 130, the screen providing an interface for the user to select from asset management options including: mapping, or identifying, new locations (e.g., servers, local and remote disks and archives) to be added to an available locations list; deleting locations from the available locations list; "copying" selected assets from selected source locations to selected destination locations; and "deleting" selected assets from selected locations as further explained below. As mentioned above, types of locations include local disks, remote disks, media servers, and archives.

The asset management GUI screen 130 includes: an available locations drop-down list 132 for displaying graphical information indicative of the list of available locations from which

the user may select a location; a server time display window 134 for displaying the local time at a selected location if the selected location, selected via the drop down list 132, is a server; and an assets list box 136 for displaying graphical information indicative of the names of assets available at the selected one of the available locations indicated in the drop down list 132, the assets in the assets list box 136 being multi-selectable so that the user may select one or more of the listed assets, the assets list box also providing for the display of graphical information indicative of the date and time that each of the listed assets was last modified.

If the selected location, selected via the drop down list 132, is a media server 18, 30 (FIG. 1), or an archive 36 (FIG. 1), then the assets list box 136 displays a list of catalogued assets. "Catalogued" assets include assets in the catalog of a media server, that is assets which are identified by the corresponding media streamer application 20 (FIG. 1) as being accessible. If the scheduled location is a disk (e.g., one of the local disks 22 or the remote disk 28 of FIG. 1), then the assets list box 36 displays all files and assets stored at the selected location. As mentioned above, media files or assets may be added from a local disk 22 (FIG. 1) to a "catalog" of a corresponding media streamer application 20 (FIG. 1), and media files or assets may be loaded from the remote disk 28 (FIG. 1) to the catalog of the corresponding media streamer application 20 (FIG. 1). Upon loading or adding a media file, which includes unparsed media data, the media streamer application 20 (FIG. 1) must parse the media data in order to convert the media file into a media asset which is formatted for staging and streaming operations.

Note that the server time window 122 is useful to the user because the GUI process is implemented as a Java applet and the user may be residing in a different time zone from the time zone which the selected server resides in. The server time window 122 allows the user to schedule actions for an appropriate time in another time zone.

The screen 130 also includes: a description display box 138 for displaying graphical information indicative of a description of the currently selected asset listed in the assets list box 136; an add/remove locations button 140 for adding locations to the list of available locations

and removing locations from the list of available locations as further explained below; a define action box 141 for defining copy actions and delete actions, the box 141 including: a copy assets button 142 for defining a copy action for "copying" a currently selected asset, indicated in the assets list box 136, from a selected source location, identified in the available locations drop down list 132; a delete selected assets button 144 for defining a "delete" action for deleting selected assets at the selected location; and a "copy to" list box 146 allowing for the user to select a destination location from a dynamically created list of possible destination locations, as further explained below.

The asset management and scheduling GUI process includes logic for determining possible destinations for a copy action from a corresponding selected source location. As mentioned above, copy actions for copying media assets include "adding" media files or assets from a local disk 22 (FIG. 1) to a catalog of a corresponding local media streamer application 20 (FIG. 1), "loading" media files or assets from a remote disk 28 (FIG. 1) to the catalog of corresponding media streamer application 20 (FIG. 1), and "staging" media assets from a source media server or a source archive to a destination media server or a destination archive. As mentioned, only media assets, as opposed to media files which include unparsed media data, may be staged. If the selected source location is a media server or an archive, then it is assumed that the copy action is a staging of the selected asset from the source location a destination location which is to either an archive or another media server. Therefore, if the selected source location is a media server or an archive, then the list of possible destination locations includes only media servers and archives. Alternatively, if the selected source location is a local disk or a remote disk, then it is assumed that the corresponding copy action is an "add" operation or a "load" operation, and the list of possible destination locations includes only media servers because if the scheduled portion of media data is a media file, then the destination media server must parse the media file in order to convert it to an asset and add it, or load it, to the catalog of the destination location.

Note that if a selected destination location is a server or other type of location requiring a log-in name and password, the user is required to enter the log-in name and password.

Note that the delete button 144 is only enabled if the selected location is a media server and the selected portion of media data is an asset. Note also that execution of a "delete" action results in a corresponding selected asset being removed from the catalog of a corresponding media streamer application 20 (FIG. 1) at the corresponding media server.

The GUI screen 130 further includes: a check box 148 for specifying delayed, or scheduled, copying wherein the selected assets are to be copied from the selected source location to the selected destination location at a later time; a start date drop down list, or drop down date picker 150 enabled by checking the check box 148 and providing for the user to select a start date for the scheduled copy action; a start time spin button time picker 152 enabled by checking box 148 and providing for the user to select a start time for the associated schedule copy action; a check box 154 for specifying that the selected asset is to be renamed before copying as further explained below; and a "define" button 155 for adding copy actions and delete actions to the action list box 156 as further explained below. If box 154 is checked, the GUI process displays a "rename-as" GUI screen (not shown) for the user to enter a target name for the asset upon copying it to the destination location.

The GUI screen 130 further includes an action list box 156 for displaying a list of copy actions and delete actions to be executed as further explained below. The action list box 156 is used for displaying graphical information indicative of a plurality of copy actions and delete actions 160. For each of the actions 160, the action list box 156 also includes: a source name field for displaying graphical information indicative of the corresponding selected asset; a target name 164 for displaying graphical information indicative of the target name, if applicable; a "from" field (not shown) for displaying the selected source location, selected using the drop down list 132; a "to" field (not shown) for displaying the selected destination location, selected via the "copy to" list box 146; and a schedule field (not shown) displaying graphical information

indicative of scheduled associated information including the start date and start time selected via components 150 and 152.

The asset management GUI 130 screen further includes: a remove button 166 for removing, or deleting, selected actions from the action list box 156; an OK button 168 for presently causing the execution of, or scheduling the execution of, the actions listed in the actions list 156; a cancel button 170 for canceling execution of the listed actions; an apply button 172 for executing or scheduling execution of the actions listed in the action list 156 without departing from the GUI screen 130; a reset button 174 for returning the GUI process to the state it was in when the user entered the screen or to the state it was in immediately after selecting the apply button (Note that no actions are sent to the server); and a help button 176 for displaying help instructions for using the GUI screen 130. Upon activation of the OK button 168, or the apply button 172, the asset management and scheduling process generates commands and associated parameters specifying the action listed in the action list 156. In the preferred embodiment, the commands and associated parameters are generated by the asset management and scheduling applet executed by the administrator terminal 12 (FIG. 1) in response to the above described user input. The commands and associated parameters are communicated to the asset management and scheduling program 23 (FIG. 1) at SERVER_1, which then executes the defined copy actions and delete actions for example, the copy actions are executed by copying the selected asset from the source location to the selected destination location on the selected date and time in accordance with the selected renaming option. Delete actions are executed after all copy actions have been executed so that selected assets are not deleted before copying.

FIG. 6 shows a flow diagram illustrating an asset management sub-process at 180. The depicted sub-process proceeds from "A" (from FIG. 4) to step 182 in which an interface is provided, the interface allowing the user to select from asset management options including: mapping, or identifying, new locations in the networked computer system 10 (FIG. 1); deleting available locations listed in the available locations drop down list 132 (FIG. 5); "copying"

selected assets from a selected source location to a selected destination location; and "deleting" selected assets from selected locations. In the preferred embodiment, the interface provided in step 182 is implemented using the asset management GUI screen 130 (FIG. 5).

From step 182, the sub-process proceeds to step 184 in which user input indicating a selected one of the asset management options is received. From step 184, the sub-process proceeds to 186 at which it is determined whether the option for mapping a new location has been selected, and if so, the sub-process proceeds to "D" (to FIG. 8A) to execute a location mapping sub-process as further explained below. The location mapping option may be selected by the user by activating the add/remove locations button 140 of the asset management GUI screen 130 (FIG. 5) which causes the asset management process to display an identify location GUI screen as further explained below.

If it is determined at 186 that the identify location option has not been selected, the sub-process proceeds to 188 at which it is determined whether the delete locations option has been selected, and if so, the process proceeds to "E" (to FIG. 9) to execute a location deleting sub-process as further explained below. Alternatively, if it is determined at 188 that the delete location has not been selected, the sub-process proceeds to 190 at which it is determined whether the option for copying and deleting assets has been selected, and if so the process proceeds to "F" (to FIG. 10A) to execute a sub-process of "copying" and "deleting" selected assets. The user may select the copy option or the delete option by activating the copy button 142 (FIG. 5) or the delete button 144 (FIG. 5) of the asset management GUI screen.

FIG. 7A shows a block diagram illustrating an identify locations GUI screen 200 for mapping new locations for storing assets. The screen 200 is displayed upon activation of the add/remove locations button 140 (FIG. 5) of the asset management GUI screen. The identify locations GUI screen 200 includes: a local disk button 202 for indicating that a local disk (e.g., local disk 22 of SERVER_1 of FIG. 1) is to be mapped; a remote disk button 204 for indicating that a remote disk (e.g., remote disk 28 of FIG. 1) is to be mapped; a media server or archives

button 206 for indicating that a media server (E.g., SERVER_1 or SERVER_2 of FIG. 1) or for indicating that an archive (e.g., archive 36 of FIG. 1) is to be mapped; a mapping path window 208 providing for entry and display of graphical information indicating a selected path to a new location; a browse button 210 for browsing predefined paths associated with corresponding ones of the buttons 202, 204, and 206; an available locations display window 212 for displaying available locations; an add button 214 enabled when the path window 208 is not blank, the add button 214 for adding a location, indicated by the path displayed in the path window 208, to the available locations list and window 212; and a remove button 216 for specifying that a location is to be deleted from the available locations window 212 and the drop down list 132 (FIG. 5).

The identify locations GUI screen 200 also includes an OK button 218 for executing the selected add and remove actions defined using the above described GUI components of screen 200, an apply button 222 for executing the add and remove actions without departing from the screen 200, and a reset button 224, each of which is enabled when the add button 214 is selected. Upon activation of the OK button 218, or the reset button 224, the newly mapped locations are added to the available location drop down list 132 (FIG. 5).

FIG. 7B shows a block diagram illustrating file transfer protocol (FTP) information interface components 227 of the identify locations GUI screen 200, the components 227 being displayed and enabled upon activation of the remote disk button 204 which indicates that a remote disk is to be mapped. The FTP information interface components 227 include: a user ID text entry box 228 providing for the user to enter a user ID for accessing a remote disk indicated by the path entered in the path window 208; and a password entry box 228 providing for the user to enter a password for accessing the remote disk.

FIG. 7C shows a block diagram illustrating a "browse local disks" GUI screen 230 for mapping a path to a local disk and directory, such as for one of the local disks 22 (FIG. 1) of the media servers 18 and 30 (FIG. 1). The "browse local disk" GUI screen 230 is activated and displayed in response to the user activating the browse button 210 (FIG. 7A) of the identify

location screen while the local disk button 202 (FIG. 7A) is activated. The "browse local disk" GUI screen 230 includes: a display window 232 for displaying graphical information indicative of folders to be mapped in a directory of a local disk, the folders being selectable by the user; an OK button 234; a cancel button 236; and a help button 238. The user may select a folder from the folders displayed in window 232, the selected folder being identified as a mapped location. Activating the OK button 234 after one of the folders has been selected causes the GUI process to map the selected folder and then return to the identify locations GUI screen 200 (FIG. 7A), and a path to the selected folder is displayed in the path window 208 (FIG. 7A). The user may then add the selected folder to the available location list box 212 (FIG. 7A) of the identify location GUI screen by activating the add button 214 (FIG. 7A). Activating the cancel button 236 cancels the users selections, if any, and returns the GUI process to the identify location GUI screen 200 (FIG. 7A) without any action. Activating the help button 238 provides graphical information for assisting the user in operating the GUI process.

FIG. 7D shows a block diagram illustrating a "browse remote drives" GUI screen 240 for mapping a path to a remote disk and directory (e.g., remote disk 28 of) FIG. 1. The "browse remote drives" GUI screen 240 is activated and displayed in response to the user activating the browse button 210 (FIG. 7A) of the identify locations screen while the remote disk button 204 (FIG. 7A) is activated. The "browse remote drives" GUI screen 240 includes: a display window 242 for displaying graphical information indicative of folders in a directory of a remote disk, the folders being selectable by the user; an OK button 234; a cancel button 236; and a help button 238. The user may select a folder from the folders displayed in window 242, the selected folder being identified as a mapped location. Activating the OK button 244 returns the GUI process to the identify locations GUI screen 200 (FIG. 7A), and a path to the selected folder is displayed in the path window 208 (FIG. 7A). The user may then add the selected folder to the available location list box 212 (FIG. 7A) by activating the add button 214 (FIG. 7A). Activating the

cancel button 246 cancels the users selections, if any, and returns the GUI process to the identify location GUI screen 200 (FIG. 7A) without any action.

FIG. 8A shows a flow diagram illustrating an identify location sub-process 250 in accordance with the present invention. The depicted sub-process proceeds from "D" (from FIG. 6) to step 252 in which an interface is provided for allowing the user to select from mapping options including: mapping a path to a local disk such as one of the local disks 22 (FIG. 1) of the media servers 18 and 30 (FIG. 1); mapping a path to a remote disk such as the remote disk 28 (FIG. 1) associated with SERVER_1 (FIG. 1); and a mapping a media server (e.g., SERVER_1 or SERVER_2 of FIG. 1) or archive (e.g., archive 36 of FIG. 1). From step 252, the sub-process proceeds to step 254 in which user input indicative of a selected one of the mapping options is received. In the preferred embodiment, the GUI screen 200 (FIG. 7A) prompts the user to select one of the buttons 202, 204, and 206 (FIG. 7A) in steps 252 and 254.

From step 254, the sub-process proceeds to 256 at which it is determined whether the local mapping option has been selected, that is whether button 202 (FIG. 7A) has been selected, and if so, the process proceeds to "D1" (to FIG. 8B). If it is determined at 256 that the local mapping option has not been selected, the process proceeds to 258 at which it is determined whether the remote mapping option has been selected, that is whether button 204 (FIG. 7A) has been selected. If it is determined at 258 that the remote mapping option has been selected, the process proceeds from 258 to "D2" (to FIG. 8C). If it is determined at 256 and 258 that neither the local mapping option nor the remote mapping option has been selected, the sub-process proceeds to 260. At 260, it is determined whether the "media server or archive mapping option" has been selected, that is whether button 206 (FIG. 7A) has been activated, and if so, the process proceeds to "D3" (to FIG. 8D). Alternatively if it is determined at 260 that the media server or archive mapping option has not been selected, the process proceeds to "D"" (back to FIG. 6).

FIG. 8B shows a flow diagram illustrating a sub-process at 270 for mapping a path to a local disk for storing media assets. The sub-process 270 proceeds from "D1" (from FIG. 8A) to

step 272 in which the user is prompted to define a path to a new local disk, and directory if applicable. From step 272, the sub-process proceeds to 274 in which information indicative of a path to the new local disk and directory is received. In the preferred embodiment, steps 272 and 274 are implemented using the identify locations GUI screen 200 (FIG. 7A) wherein the user provides user input indicating a path to a new local disk by entering text indicative of a path in the path window 208 (FIG. 7A). Also in the preferred embodiment, the user may optionally activate the browse button 210 (FIG. 7A) which causes the GUI process to display the browse local disks screen 230 (FIG. 7C) which assists the user in indicating a path to a new local disk as explained above. From step 274, the sub-process proceeds to step 276 in which a new available location defined by the path is added to an available location list. From step 276, the sub-process proceeds to step 278 in which a revised available locations list is displayed. In the preferred embodiment, the revised available locations list is displayed in the available locations window 212 (FIG. 7A).

FIG. 8C shows a flow diagram illustrating a sub-process at 280 for mapping and identifying a new remote disk as one of the available locations. The depicted sub-process proceeds from "D2" (from FIG. 8A) and proceeds to step 282 in which the user is prompted to enter: a path to a new remote disk and directory; and a user ID and password for accessing the new remote disk if required. From step 282, the sub-process proceeds to step 284 in which user input indicative of a path, a user ID, and a password is received. From step 284, the sub-process proceeds to step 286 in which a new remote location is added to the available location list. In step 288, a revised location list is displayed. From step 288, the process proceeds back to step "D'" (back to FIG. 8A). In the preferred embodiment, the sub-process 280 is implemented using the identify locations GUI screen 200 (FIG. 7B).

FIG. 8D shows a flow diagram illustrating a sub-process at 290 for defining a path to a media server, such as one of the media servers 18 and 30 (FIG. 1), or an archive such as the archive 36 (FIG. 1). The sub-process 290 proceeds from "D3" (from FIG. 8A) to step 292 in

which the user is prompted to enter a path to a new media server or archive location. From step 292, the sub-process proceeds to step 294 in which user input indicative of a path to a new media server or archive location is received. In the preferred embodiment, steps 292 and 294 are implemented using the identify locations GUI screen 200 (FIG. 7A) wherein the user provides user input indicating a path to a new media server or archive location by entering information in the parameter path window 208 (FIG. 7A). From step 294, the sub-process proceeds to step 296 in which the path to the new remote location specified in step 294 is added to the available locations list displayed in the available locations window 212 (FIG. 7A) by activating the add button 214 (FIG. 7A). In step 298, the revised available location list is displayed in window 212 (FIG. 7A). From 298, the sub-process proceeds to "D3'" (back to FIG. 8A). Note that a user of the asset management and scheduling GUI process at the administrator terminal must be authorized to access locations such as servers. Particular locations (e.g., SERVER_1, archive 36, and SERVER_2) may require a log-in name and password in order to access the location. The user is required to enter a log-in name and password upon commiting/defining an action to a server.

FIG. 9 shows a flow diagram illustrating a sub-process at 302 for deleting selected locations from the available locations list displayed in window 212 (FIG. 7A) of the identify locations GUI screen 200. The sub-process 302 proceeds from "E" (from FIG. 6) to step 302 in which a list of available locations is displayed. In the preferred embodiment, the available locations list is displayed in window 212 (FIG. 7A) of the identified location GUI screen. From step 302, the sub-process proceeds to step 304 in which an interface is provided allowing the user to select locations to be removed from the available locations list. From step 304, the sub-process proceeds to step 306 in which user input indicating a selected one of the locations to be removed is received. From step 306, the sub-process proceeds to step 308 in which the selected location, or locations, are removed from the available locations list. In step 310, a revised available locations list is displayed. In the preferred embodiment, the sub-process 302 is

implemented using the identify locations GUI screen 200 (FIG. 7A) wherein the user selects one or more locations from the locations displayed in the window 212 (FIG. 7A) and activates the remove button 216 which causes the selected locations to be removed from the list, and window 212 (FIG. 7A) then displays the revised available locations list. From step 310, the process proceeds back to "E"' (back to FIG. 6).

FIG. 10A shows a flow diagram illustrating a sub-process at 320 for "copying" and "deleting" selected assets. The sub-process 320 proceeds from "F" (from FIG. 6) to step 322 in which an interface allowing the user to select from a displayed list of available locations is provided. The displayed list includes: locations of local disks such as local disk 22 (FIG. 1); locations of remote disks such as remote disk 28 (FIG. 1); locations of media servers such as SERVER _1 or SERVER _2 (FIG. 1); and locations of archives such as archive 36 (FIG. 1). Also in step 322, user input indicating a selected one of the locations is received. In the preferred embodiment, the sub-process 320 of is implemented using the asset management GUI screen 130 (FIG. 5). Therefore, in the preferred embodiment, step 322 is implemented using the available locations drop down list 132 (FIG. 5) which provides the list of available locations from which the user may select as described above.

Note that upon selecting a location, if the user has not already logged on to the server associated with the selected location, the asset management and scheduling process executes instructions at the user terminal 12 (FIG. 1) for accessing the selected server associated with the selected location (e.g., SERVER_1 or SERVER_2 of FIG. 1). Note that the user may be required to enter a log-in name and password in order to establish a connection with the selected server.

From step 322, the sub-process proceeds to 324 at which it is determined whether the location selected in step 322 is a server, and if so, the sub-process proceeds to step 326 in which the local time at the selected server is displayed. In the preferred embodiment, the local time of the selected server is displayed in the server time window 134 (FIG. 5) of the asset management

GUI screen. As mentioned above, the server time window 122 is useful to the user because the GUI process is implemented as a Java applet and the user may be residing in a different time zone from the time zone which the selected server resides in. The server time window 122 allows the user to schedule actions for an appropriate time in another time zone.

From step 326, the sub-process proceeds to step 328 in which a list of assets located at the selected location is displayed along with a corresponding date and time that each of the displayed assets were last modified. In the preferred embodiment, these assets are displayed in the assets window 136 (FIG. 5) of the asset management GUI screen. In step 330, an interface allowing the user to select from the list of assets located at the selected location is provided, and user input indicating a selected one of the assets is received. In the preferred embodiment, the user may select one of the assets in the assets list box 136 (FIG. 5) by clicking on it. Note that the assets list box 136 (FIG. 5) is multi-selectable.

In step 332, a description of the selected one of the assets located at the selected location is displayed. In the preferred embodiment, the description of the selected asset is displayed as graphical information in the description window 138 (FIG. 5) of the asset management GUI screen. From step 332, the sub-process proceeds to step 334 in which an interface is provided, the interface allowing the user to select from copy/delete options including "copying" the selected asset, and "deleting" the selected asset. Also in step 334, user input indicating a selected one of the copy/delete options is received. In the preferred embodiment, the user may select the option of copying selected assets by activating the copy button 142 (FIG. 5) of the asset management screen, or the user may select the delete selected assets option by activating the delete button 144 (FIG. 5) of the asset management GUI screen. The delete option is only possible if the selected location, selected via the drop down list 132 (FIG. 5), is a media server. As mentioned above, the execution of the delete action does not actually delete the selected asset from the selected location, but rather removes the selected asset from the catalog of the corresponding media streamer application 20 (FIG. 1). The user of the asset management and

scheduling GUI generally does not have authority to actually delete assets. Generally, only a local administrator of the corresponding server may actually delete files.

From step 334, the sub-process proceeds to 336 at which it is determined whether the delete assets option has been selected, and if so, the GUI process proceeds to "F1" (to FIG. 10B) to execute a sub-process for deleting a selected asset from a selected location as described below. Alternatively, if it is determined at 336 that the delete assets option has not been selected, the sub-process proceeds to 338 at which it is determined whether the copy option has been selected, and if so, the process proceeds to "F2" (to FIG. 10C). If it is determined at 336 and 338 that neither the copy option nor the delete option has been selected, the process proceeds back to "F'" (back to FIG. 6).

FIG. 10B shows a flow diagram illustrating a sub-process at 342 for defining a delete action for "deleting" a selected asset from a selected location. The sub-process 342 proceeds from "F1" (from FIG. 10A) to step 344 in which a delete action, associated with instructions for deleting the currently selected asset from the selected location, is defined. From step 344, the sub-process proceeds to step 346 in which the delete action defined in step 344 is added to an action list. In step 348, an updated action list is displayed after which the process proceeds back to "F1'" (back to FIG. 10A). In the preferred embodiment, the sub-process 342 is implemented using the asset management GUI screen 130 (FIG. 5) wherein the user may delete a selected asset, which has been selected via the assets window 136 (FIG. 5), by activating the delete button 144 (FIG. 5) of the asset management GUI screen 130, and then activating the define button 158 (FIG. 5) which adds the defined delete action to the action list box 156 (FIG. 5).

FIG. 10C shows a flow diagram illustrating a sub-process at 350 for "copying" a selected asset from a selected source location to a selected destination location. The sub-process 350 proceeds from "F2" (from FIG. 10A) to step 352 in which the user is prompted to select a destination location. Also in step 352, user input indicating the selected destination location is received, the source location having been selected in step 322 (FIG. 10A). In the preferred

embodiment, sub-process 350 is implemented using the asset management GUI screen 130 (FIG. 5) wherein the user selects the destination location from the list of possible destination locations listed in the "copy-to" window 146 (FIG. 5) as explained above.

Note that upon selecting a destination location, if the user has not already logged on to the server associated with the selected destination location, the asset management and scheduling process executes instructions at the user terminal 12 (FIG. 1) for accessing the selected server associated with the selected destination location (e.g., SERVER_1 or SERVER_2 of FIG. 1). At this point, the user may be required to enter a log-in name and password in order to establish a connection with the selected destination server. A connection with the selected destination server is required in order to verify that the user has authority to copy media data to the destination media server.

From step 352, the sub-process proceeds to step 354 in which an interface is provided, the interface allowing the user to select from timing options including an immediate copying option, and a delayed copying option wherein selected assets are scheduled for copying at a specified later time. Also in step 354, user input indicating the selected timing option is received. From step 354, the sub-process proceeds to 356 at which it is determined whether the timing option of copying a selected asset at a specified later time has been selected. In the preferred embodiment, the user may select displayed copying option by checking the "copy assets later" box 148 (FIG. 5).

If the delayed copying option is selected, the sub-process proceeds from 356 to step 358 in which an interface allowing the user to select a start date and start time is provided. Also at step 358, user input including scheduling information associated with the copy action is received, the scheduling information including a selected start date and start time. In the preferred embodiment, the user selects the start date and start time using the start date drop down window 150 (FIG. 5) and the start time spin button 152 (FIG. 5) respectively. If the delayed copying

option is not selected, then it is assumed that scheduling information associated with the defined copy action specified that the copy action is to be executed presently.

From step 358, the sub-process proceeds to step 360 in which an interface, allowing the user to select an option of renaming the asset before copying, is provided. Also in step 360, user input indicating a selected renaming option is received. In the preferred embodiment, the user may select the option of renaming assets before copying by checking box 154 (FIG. 5) of the asset management GUI screen 130 (FIG. 5). From step 360, the process proceeds to "F3" (to FIG. 10D) to execute further steps of the copy/delete selected assets sub-process.

FIG. 10D shows a flow diagram at 362 illustrating further steps of the copy/delete selected assets sub-process 320 (FIG. 10A). Sub-process 362 proceeds from "F3" (from FIG. 10C) to 364 at which it is determined whether the renaming option has been selected, and if so, the sub-process proceeds to step 366 in which an interface allowing the user to enter a target name for the copied asset is provided. Also in step 366, user input indicating a target name is received. In the preferred embodiment, the user provides input indicating a target name for the selected asset using a rename asset GUI screen 390 shown in FIG. 11. With reference to FIG. 11, the rename asset GUI screen 390 includes a current name window 392 for displaying a current name of the selected asset; a "rename as" window 394 allowing for the user to enter a target name for the selected asset; an OK button 396; a cancel button 398; and an help button 400.

Referring back to FIG. 10D, the sub-process proceeds from 366 to step 368 in which a copy action is defined, the copy action being associated with instructions for copying the currently selected asset, or assets, from the selected source location to the selected destination location in accordance with the specified scheduling information (at the selected date and time if the delayed copying option has been selected, or presently if the delayed copy option has not been selected) in accordance with the selected renaming option. In the preferred embodiment,

the copy action is defined using the GUI components of the asset management GUI screen 130 (FIG. 5) as explained above.

From step 368, the sub-process proceeds to step 370 in which the copy action defined in step 368 is added to an action list, and then to step 372 in which an updated action list is displayed. In the preferred embodiment, copy actions are added to the action list by activating the define button 155 (FIG. 5) of the asset management GUI screen which adds the defined copy action to the action list displayed in window 156 (FIG. 5). In step 374, an interface allowing the user to execute the actions listed in the action list is provided, and user input is received. In the preferred embodiment, the user may execute the actions listed in the action list 156 (FIG. 5) of the asset management GUI screen by activating the OK button 168 (FIG. 5), or the apply button 172 (FIG. 5), after actions have been defined.

From step 374, the sub-process proceeds to 376 at which it is determined whether the option of executing the defined actions has been selected. It is important that all copy actions associated with a particular asset be executed before any delete actions associated with the particular asset so that the particular asset is not deleted before it may be copied. If it is determined at 376 that the defined actions are to be executed, the sub-process proceeds to step 380 in which, for each listed copy action, a set of copy commands and associated parameters is generated, each set of copy commands and associated parameters specifying that the corresponding selected asset is to be "copied" from the corresponding selected source location to the corresponding selected destination location in accordance with the corresponding selected renaming option and in accordance with the specified scheduling information. From step 380, the sub-process proceeds to step 382 in which, for each listed delete action, a set of delete commands and associated parameters is generated, each set of delete commands and associated parameters specifying that the corresponding selected asset is to be "deleted" from the corresponding selected source location as defined by the corresponding delete action.

From step 382, the sub-process proceeds to step 384 in which each set of copy commands and associated parameters, and each set of delete commands and associated parameters is transmitted to the corresponding media server (e.g., SERVER_1 or SERVER_2 of FIG. 1) which is identified by the corresponding source location. In response to the sets of commands and associated parameters, the media streaming application 20 (FIG. 1) of the corresponding media server is operative to schedule execution of, or presently execute, the corresponding copy and delete operations as defined by the actions specified by the commands and associated parameters. For example, the media streamer application 20 (FIG. 1) of the media server associated with the selected source location is operative to copy the corresponding selected assets from the corresponding selected source location to the corresponding selected destination location in accordance with the selected renaming option and the specified scheduling information. In one embodiment, the asset management and scheduling GUI process, executed at the administrator terminal, transmits the sets of commands and associated parameters to the corresponding media servers via the IP network 16 (FIG. 1). Note that connections with the selected source locations and destination locations were previously made at which time user names and passwords were provided, if required. The connections with the locations need not be continuously maintained but may be re-established when required. From step 384, the process proceeds back to "F"" (back to FIG. 6).

FIG. 12 shows a block diagram illustrating an encoding GUI screen 420 for encoding a new media asset as further explained below. The GUI screen 420 is displayed to the user at the administrative terminal 12 (FIG. 1) when the "encode a new asset icon" 104 (FIG. 3) is selected. The GUI screen 420 includes: an encoder drop down list 422 allowing the user to select an encoder for encoding a new asset (e.g., SERVER_1 which is connected to camera 24 of FIG. 1); an encoder time display box 424 for displaying the local time at the selected encoder which is a selected one of the media servers; an encoder attributes button 426 for selecting audio and video encoding attributes for the encoding process as further explained below; a preview button 428 for

activating a browser plug-in at the administrator terminal 12 (FIG. 1) for displaying a scene to be encoded by the selected encoder; and a duration window 430 for defining schedule information specifying a schedule for the encoding operation as further explained below, the GUI screen 420 also provides for the user to define and schedule streaming operations, recording operations, and playback operations to be performed on the newly encoded asset.

The duration window 430 includes: a manual option button 432 for specifying manual mode operation which allows for the user to manually start and stop encoding as further explained below; and a timed encoding option button 434 for specifying scheduling information associated with the encoding operation, the scheduling information indicating when the encoding operation is to start and stop. The duration window 430 also includes GUI components associated with the timed encoding mode of operation, the components including: a start group box 435 having a start-now button 436 for specifying that encoding is to start immediately, and a schedule time button 438 which enables a start date drop down date-picker 440 and a start time spin button 442 further explained below; and a stop group box 443 having a duration button 444 which enables a duration spin button and window 446, and a schedule time button 448 which enables a start date drop down window 450, and a start time spin button 452, further explained below. Note that when the manual option button 432 is activated, the contents of the start and stop group boxes 435 and 443 are disabled. Also, activating the duration button 444 disables the start date and start time GUI components 450 and 452. When the timed encoding button 434 is activated, the user is prompted to provide scheduling information via the start and stop group boxes 435 and 443. The timed encoding may be scheduled to begin immediately by activating the start now button 436, or at a scheduled date and time by selecting via components 440 and 442. The timed encoding may be scheduled to stop either after a specified duration selected via the duration window 446, or at a specified time and date selected via the GUI components 450 and 452.

The encoding GUI screen 420 further includes: a streaming check box 454 for specifying that the asset is to be streamed from the encoder to one or more selected end users in real time that is, within a minimal time delay after which the asset is encoded; and a stream box 455 having interface components which are enabled upon checking box 454. The stream box 455, which allows for the user to specify destination end user terminals to which newly encoded asset is to be streamed, includes: a manual option button which enables the user to specify one destination IP address and port number via an IP address text box 462, and a port number text box 464; a predefined option button 458 which enables the user to select from predefined groups of destination IP addresses and port numbers listed in a drop down list 466, the option button 458 also enabling a define button 468 which allows the user define a new group of destination IP addresses and port numbers as further explained below; and a session director requester (SDR) option button 460 which is enabled if the user has third party software. In response to the user providing a start time and duration, SDR determines an available IP address and port number.

The encoding GUI screen 420 further includes: a record asset check box 470 for specifying that an asset is to be recorded as it is encoded; and a record asset box 471 having interface components which are enabled upon checking box 470. The record asset box 471 includes: a "record to" drop down combination box 472 allowing the user to select a record path from previously defined record paths, the selected record path specifying a selected "record to" memory device and directory to which the asset is to be stored; a browse button 474 which displays the "browse local disks" GUI screen 230 (FIG. 7C) for defining a path for a local disk and directory, such as for the local disk 22 of SERVER_1 (FIG. 1); an asset name text box 476 providing for the user to enter text indicative of a name (e.g., "Trial.mpg") for the new asset; a description text box 478 providing for the user to enter textual information describing the new asset; and three overwrite option buttons explained below.

The three file option buttons include: a prompt option button 480 which, when activated, specifies that the user is to be prompted if an asset having the name indicated in the asset name

text box 476 already exists at the selected "record to " memory device; an overwrite option button 482 which, when activated, specifies that any asset having the same name as that indicated in the asset name text box 476 which exists a the "record to" memory device is to be overwritten with the new encoded asset; and an append option button 484 which, when activated, specifies that any asset having the same name as that indicated in the asset name text box 476 is to be appended with the encoded and parsed media data comprising the new asset.

The encoding GUI screen 420 further includes: a schedule playback check box 486, enabled if the manual option button 432 is not selected for scheduling a play-back (streaming or multicasting) operation to be executed on the newly encoded asset in accordance with a play-back schedule. The check box 486 enables a schedule play-back button 488 for activating GUI components for specifying a play-back schedule as further explained below; an OK button 490; a cancel button 492; and a help button 494.

FIG. 13A shows a flow diagram illustrating an encoding sub-process at 500. The depicted sub-process proceeds from "B" (from FIG. 4) to step 502 in which an interface, allowing the user to select a source encoder, is provided. Also in step 502, user input indicating a selected source encoder is received. In the preferred embodiment of the present invention, the encoding sub-process 500 is implemented using the encoding GUI screen 420 (FIG. 12). From step 502, the sub-process proceeds to step 504 in which graphical information indicative of the selected source encoder is displayed, and local time at the location of the selected source encoder is also displayed. In the preferred embodiment, steps 502 and 504 are implemented using the encoder drop down list 422 (FIG. 12), and the server time window 424 (FIG. 12).

From step 504, the depicted sub-process proceeds to step 506 in which an interface allowing the user to select the option of changing attributes of the selected encoder is provided, and user input indicative of the selected option is received. In the preferred embodiment, step 506 is implemented using the encoder attributes button 426 (FIG. 12) of the encoding GUI screen. From step 506, the sub-process proceeds to 508 at which it is determined whether the

option of changing attributes of the selected encoder has been selected, and if so, the process proceeds to "B1" (to FIG. 15) to execute a sub-process of defining encoder attributes.

If it is determined at 508 that the option of changing encoder attributes has not been selected, the sub-process proceeds to step 510 in which an interface allowing user to select a preview option, for previewing a scene to be encoded, is provided. Also in step 510, user input indicative of the selected preview option is received. In the preferred embodiment, step 510 is implemented using the preview button 428 (FIG. 12) of the encoding GUI screen. From step 510, the sub-process proceeds to step 512 at which it is determined whether the preview option has been selected, and if so, the sub-process proceeds to step 513 in which a browser plug-in, showing a scene to be encoded, is displayed. The browser plug-in may be implemented in accordance with any of the variety of well known methods. In step 513, preview commands are transmitted from the administrator terminal 12 (FIG. 1) to the selected encode via the IP network. In response to the preview commands, the selected encoder generates media data associated with a scene to be encoded, and streams the media data back to the administrator terminal. For example, if the selected encoder specifies SERVER_1 (FIG. 1) and camera 24 (FIG. 1), then the camera 24 is activated by SERVER_1 in response to preview commands sent to SERVER_1, and media data is generated by camera 24 (FIG. 1) and transmitted to the administrator terminal 12 (FIG. 1) at which the media data is displayed to the user in a plug-in displayed within a browser window on the display unit 62 (FIG. 2) of the administrator terminal.

From step 513, the sub-process proceeds to step 514 in which an interface allowing the user to select from encoding options is provided. The encoding options include a manual encoding option, and a timed encoding option. Also in step 514, user input indicating a selected encoding option is received. In the preferred embodiment, step 514 is implemented using the manual option button 432 (FIG. 12) and the timed encoding option button 434 (FIG. 12) of the encoding GUI screen. From step 514, the sub-process proceeds to 516 at which it is determined whether the manual encoding option has been selected, and if so, the process proceeds to "B2"

(to FIG. 17). Alternatively, if it is determined at 516 that the manual encoding option encoding has not been selected, the sub-process proceeds to step 518 in which an interface, allowing the user to specify scheduling information for timed encoding is provided. Also in step 518, user input including scheduling information for a timed encoding operation is received. In the preferred embodiment, step 518 is implemented using the GUI components of the start group box 435 (FIG. 12) and the stop group box 443 (FIG. 12) of the encoding GUI screen as described above.

FIG. 13B shows a flow diagram illustrating further steps of the encoding sub-process 500 (FIG. 13A). The sub-process proceeds from "B3" (from FIG. 13A) to step 526 in which: an interface allowing the user to select a real time streaming option is provided; and user input indicative of the selected streaming art option is received. In the preferred embodiment, step 526 is implemented using the streaming check box 454 (FIG. 12) of the encoding GUI screen. From step 526, the sub-process proceeds to step 528 at which it is determined whether the real time streaming option has been selected, and if so, the process proceeds to "B4" (to FIG. 18) to execute a sub-process of defining destination locations to which the encoded asset is to be streamed in real time.

Alternatively, if it is determined at 528 that the real time streaming option has not been selected, the sub-process proceeds to step 530 in which: an interface allowing the user to select a record asset option is provided; and also user input, indicative of whether the record asset option has been selected, is received. In the preferred embodiment, step 530 is implemented using the record asset option check box 470 (FIG. 12) of the encoding GUI screen. From step 530, the sub-process proceeds to 532 at which it is determined whether or not the recording option has been selected, and if so, the process proceeds to "B5" (to FIG. 21) to execute a sub-process of specifying a location to which the newly encoded asset is to be stored, as well as providing other recording options including naming the encoded asset, providing a description of the encoded asset, and specifying overwrite options for recording the asset, all of which are described below.

Alternatively, if it determined at 532 that the record asset option has not been selected, the process proceeds to step 536 in which commands and parameters associated with scheduled encoding operations, streaming operations (if specified), and recording operations (if specified) are generated. Scheduled encoding commands and associated parameters are generated based on encoding operation information and encoding scheduling information specified by the user via the encoding GUI components 422 through 452 (FIG. 12) of the encoding GUI screen. Streaming commands and associated parameters are generated based on real time streaming operation information specified by the user via the GUI components of the streaming group box 455 (FIG. 12) of the encoding GUI screen. Recording commands and associated parameters are generated based on recording operation information specified by the user via the GUI components 470 through 484 (FIG. 12) of the encoding GUI screen.

From step 536, the process proceeds to step 538 in which the scheduled encoding commands and associated parameters (as well as the streaming and recording commands and associated parameters, if applicable) are transmitted to the selected source encoder. In response to the received commands, the selected source encoder, which is a selected one of the media servers 18, 30 (FIG. 1) executes the specified scheduled encoding operations (as well as the recording and real time streaming operation, if selected, in accordance with the specified operation information and schedule information. From step 538, the process proceeds to "B"" (back to FIG. 4).

FIG. 14 shows a block diagram illustrating an encoder attributes defining GUI screen 550 in accordance with the present invention. The screen 550 is displayed upon activation of the encoder attributes button 426 (FIG. 12) of the encoding GUI screen 420 (FIG. 12). The GUI screen 550 includes: a logical port name drop down list 552 allowing the user to select an alias name for the selected encoder; an encoder type display window for displaying graphical information indicative of a type of the selected encoder, the window 554 being non-editable; an encoder quality drop down list 556 allowing the user to select an encoding quality such as

MPEG1, MPEG2, and other video encoding types; and a set bit rate box 557 for specifying a bit rate for an encoding operation.

The set bit rate box 557 includes: a mux rate option button 558 which enables a low bit rate option button 564 (0.8 Mbps), a medium bit rate option button 566 (1.5 Mbps), and a high bit rate option button 568 (2.2 Mbps); and a video and audio option button 560 which enables a video label and text box 570, and an audio drop down list 572.

The VideoCharger media server offers a solution for both the Internet environments as well as the Intranet. For the Internet clients, especially the home Internet users which are typically connected via slower network connections, VideoCharger will support the delivery of Low Bit Rate video (LBR). The LBR video is based on the Industry Standard H.263 video and G.723 audio from the video conferencing industry. This technology allows audio and video to be served to home Internet users connected with 28.8 Kb modems. Using ~ 16 Kb/s, the LBR video will offer 8 KHz 16 bit PCM audio and 160X120 video at 7.5 frames per second. The LBR video can be encoded at higher quality rates to provide higher resolution or more frames per second for those clients which are connected via ISDN modems, cable modems or an Intranet network. For the Intranet environment, the VideoCharger media server provides support for higher quality (and higher bit rate) videos. In this environment, both MPEG-1 and MPEG-2 content can be supported at rates up to 8 Mb/s. Near CD quality single-channel audio is supported at 24KHz. This flexible support for higher quality video will allow a multitude of applications to be enhanced with video in the Intranet environment. VideoCharger includes transcoders that can convert either Windows .AVI or .WAV files into low bit-rate format. The compressed data is stored in a unique file format (.IBA) optimized for UDP protocol.

The GUI screen 550 further includes: an advanced box 574 having a mux mode drop down list 576 for specifying how to combine video and audio data, a video source drop down list 578, and a group of pictures (GOP) spin button 580 providing for the user to select from groups

of pictures between headers. The GUI screen 550 further includes: an OK button 582 for making everything okay; a cancel button 584; and a help button 586.

FIG. 15 shows a flow diagram illustrating a sub-process for defining encoder attributes at 600. The depicted sub-process proceeds from "B1" (from FIG. 13A) to step 602 in which an interface, allowing the user to select from a list of alias names (logical port names) for the selected encoder is provided. Also in step 602, user input indicated of the selected alias name is received. In the preferred embodiment, the sub-process 600 is implemented using the encoder attributes GUI screen 550 (FIG. 12). Specifically, step 602 is implemented using the logical port name drop down list 552 (FIG. 12). From step 602, the sub-process proceeds to step 604 in which an interface, allowing the user to select from the list of quality types for the selected encoder, is provided. Also in step 604, user input indicative of a selected quality type is received. In the preferred embodiment, step 604 is implemented using the encoder quality drop down list 556 (FIG. 14).

From step 604, the sub-process proceeds to step 606 in which an interface, allowing a user to select from bit rate options including a mux rate option and a video/audio option, is provided. Also in step 606, user input indicative of a selected bit rate option is received. In the preferred embodiment, step 606 is implemented using the set bit rate box 557 (FIG. 14) in which the user may select the mux rate option button 558 or the video/audio option button 560 (FIG. 14). From step 606, the sub-process proceeds to step 608 at which it is determined whether the mux rate option has been selected, and if so, the sub-process proceeds to step 610 in which an interface allowing the user to select a mux rate, is provided. Also in step 610, user input indicative of the selected mux rate is received. In the preferred embodiment, step 610 is implemented using the low, medium, and high buttons 564, 566, and 568 (FIG. 14) of the GUI screen 550.

Alternatively, if it is determined at 608 that the mux rate option has not been selected, the sub-process proceeds directly to step 612 at which it is determined whether the video/audio

option has been selected. If the video/audio option has been selected, the sub-process proceeds to step 614 in which an interface, allowing the user to enter a video bit rate and an audio bit rate value is provided. Also in step 614, user input indicative of a selected video bit rate and a selected audio bit rate is received. In the preferred embodiment, step 614 is implemented using the video rate text box 570 and the audio rate drop down list 572 (FIG. 14). From step 614, the process proceeds to "B1'" (back to FIG. 13A).

FIG. 16 shows a block diagram illustrating a GUI screen 620 for manually encoding a new asset. The GUI screen 620 is activated, and displayed upon selection of the manual option button 432 (FIG. 12) of the encoding asset GUI screen. The manual encoding GUI screen 620 provides an interface allowing for the user to remotely start, pause, and stop encoding of video data at the selected encoder, selected via the encoder drop down list 422 (FIG. 12). In accordance with the GUI process of the present invention, manual encoding commands and associated parameters are generated at the administrator terminal 12 (FIG. 1) and transmitted to the selected encoder via the IP network 16 (FIG. 1). In response to the manual encoding commands, the media streamer application 20 (FIG. 1) of the encoder is operative to start, stop, and pause encoding of media data received from the video camera.

The manual encoding GUI screen 620 includes: a record button 622 for starting encoding of the asset; a pause button 624 for pausing the encoding of the asset; a stop button 626 for stopping the encoding of the asset; an elapsed time window 628 for displaying an elapsed time clock which is running while the asset is being encoded, the elapsed time clock being started, paused, and stopped upon activation of buttons 622, 624, and 626 respectively; a "stream to" window 630 for displaying an IP address and port number to which the encoded asset is being streamed in real time, if applicable; a "record to" window 632 for displaying a path of a location to which the encoded asset is being is recorded to, if applicable; an asset name window 634 for displaying an asset name specified for the encoded asset; a description window 636 for

displaying a description of the encoded asset; and an encoder window 638 for displaying a name indicative of the selected encoder.

FIG. 17 shows a flow diagram illustrating a sub-process at 50 of manually and remotely controlling a process of encoding an asset. The sub-process 50 proceeds from "B2" (from FIG. 13A) to step 652 in which a browser plug-in showing a scene to be encoded is displayed. In the preferred embodiment, the sub-process 650 is implemented using the manual encoding GUI screen 620 (FIG. 16) which is displayed in a browser window displayed on the display device of the administrator terminal 12 (FIG. 1).

From step 652, the sub-process proceeds to step 654 in which text indicating details is displayed. The displayed details include an elapsed time, a destination IP address and port number to which the encoded data is to be streamed in real time, a record storage path location for storing the encoded asset, the asset name, an asset description, and a selected encoder. In the preferred embodiment, step 654 is implemented using the GUI screen 620 which displays the details 628 through 638 (FIG. 16). From step 654, the sub-process proceeds to step 656 in which a user interface, allowing a user to select from manual encoding commands including start/resume encoding, pause encoding, and stop encoding is provided. Also in step 656, a user input indicative of a selected one of the manual encoding commands is received.

From step 656, the sub-process proceeds to 658 at which it is determined whether the start manual encoding option has been selected, and if so, the sub-process proceeds to step 660 in which commands for starting encoding of video data are generated and communicated from the administrator terminal 12 (FIG. 1) to the selected encoder via the IP network (FIG. 1). From step 660, the sub-process proceeds to step 662 in which an elapsed time clock is started . In the preferred embodiment, the elapsed time is displayed in the elapsed time window 628 of the manual encoder GUI screen 620 (FIG. 16).

If it is determined at 658 that the start encoding option has not been selected, the sub-process proceeds to 664 at which it is determined whether the pause encoding option has

been selected. If it is determined at 664 that the pause encoding option has been selected, the sub-process proceeds to step 668 in which commands for pausing the process of encoding video data are generated and communicated from the administrator terminal 12 (FIG. 1) to the selected encoder via the IP network (FIG. 1). From step 668, the sub-process proceeds to step 670 in which the elapsed time clock is paused .

If it is determined at 664 that the pause encoding option has not been selected, the sub-process proceeds to 672 at which it is determined whether the stop encoding option has been selected. If it is determined at 672 that the stop encoding option has been selected, the sub-process proceeds to step 674 in which commands for stopping the encoding of video data are generated and communicated from the administrator terminal 12 (FIG. 1) to the selected encoder via the IP network (FIG. 1). From step 674, the sub-process proceeds to step 676 in which the elapsed time clock is stopped .

FIG. 18 shows a flow diagram illustrating a streaming sub-process at 700 in accordance with the present invention. The streaming sub-process 700 proceeds from "B4" (from FIG. 13B) to step 702 in which an interface allowing the user to select from streaming destination specification options is provided. The destination specification options include a manual option, a predefined option, a define option, and a session director request option. Also in step 702, user input indicative of the selected destination specification option is received. In the preferred embodiment, step 702 is implemented using the IP address window 462 (FIG. 12) and the port number drop down list 464 (FIG. 12). In the preferred embodiment, the sub-process 700 is implemented using the encoding GUI screen 420 (FIG. 12), and in particular, the GUI components of the stream out group box 455 which is enabled by the stream out check box 454 (FIG. 12). The manual option, predefined option, define option, and SDR option may be selected by the user using the manual option button 456, predefined option button 458, define button 468, and SDR button 460 (FIG. 12) respectively.

From step 702, the depicted sub-process proceeds to 704 at which it is determined whether the manual option has been selected, and if so, the sub-process proceeds to step 706 in which an interface, allowing the user to select a destination IP address and port number, is provided. Also in step 706, user input indicative of the selected destination IP address and selected destination port number is received. From step 706, the sub-process proceeds to 708 at which it is determined whether the predefined option has been selected. Also if it is determined at 704 that the manual option has not been selected, the sub-process proceeds directly from 704 to 708. If it is determined at 708 that the predefined option has been selected, the sub-process proceeds to step 710 in which a user interface, allowing the user to select a group from a list of predefined groups of destination IP addresses and port numbers, is provided. Also in step 710, user input indicative of selected groups is received. In the preferred embodiment, step 710 is implemented using the GUI component 466 (FIG. 12) described above.

From step 710, and also from 708 in the case that the predefined option has not been selected, the sub-process proceeds to 712 at which it is determined whether the define option has been selected, and if so, the sub-process proceeds to "B41" (to FIG. 20). If is determined at 712 that the define option has not been selected, the sub-process proceeds to 714 at which it is determined whether or not the SDR option has been selected, and if so, the process proceeds to "B42" (to FIG. ?). Alternatively, if it is determined at 714 that the SDR option has not been selected, the process proceeds to step 715 in which session director request software is executed after which the process proceeds to "B4(" (back to FIG. 12B).

FIG. 19 shows a block diagram illustrating a GUI screen 720 for defining a destination group of IP addresses and port numbers. The define group GUI screen 720 includes: a group name text box 722 providing for the user to enter a name for set of IP addresses and port numbers; a delete button 724 which is enabled if there are previously defined groups; and a define group box 725.

The define group box 725 includes: a destination group box 726 having an IP address text box 727 providing for the user to enter one destination IP address, the text box 727 enabling an add button as further explained below, and a port number text box 728 having a spin button providing for the user to select a destination port number, the spin button allowing the user to select values limited to a valid range of port numbers reserved for a multicast operation; an add button 730 for adding the IP address and port number specified in text boxes 727 and 728 to a member list, the add button resetting the IP address text box 727 to be blank, thereby disabling the add button; a remove button 732 allowing the user to remove selected members from the member list; a member list box 734 for listing IP addresses and port numbers for each member in a format "123.123.123.123.123;5001"; an OK button 736 for adding a group of IP addresses and port numbers to the list of predefined groups displayed in the drop down list 466; a cancel button 738; and a help button 740. The member list box 734 is multi-selectable (shift-contiguous selection; control-disjoint selection). The remove button 732 removes the selected row of IP addresses and port number from the number list. If the member list is empty, the remove button 732 is disabled.

FIG. 20 shows a flow diagram illustrating a sub-process at 750 for defining a destination group of IP addresses and port numbers for the purpose of defining a streaming operation. The sub-process 750 proceeds from "B41" (from FIG. 18) to step 752 in which a user interface allowing the user to enter a destination group name for a set of IP addresses and port numbers is provided. Also in step 752, user input indicating a selected destination group name is received. In the preferred embodiment, the sub-process 750 is implemented using the define group GUI screen 720 (FIG. 19). Step 752 is implemented using the group name text box 722 (FIG. 19). From step 752, the sub-process proceeds to step 754 in which an interface allowing the user to select from a list of previously defined destination group names for the purpose of deleting a selected group name, is provided. Also in step 754, user input indicative of a selected destination

group name is received. In the preferred embodiment, step 754 is implemented using the delete button 724 (FIG. 19).

From step 754, the sub-process 750 proceeds to 756 at which it is determined whether the delete option has been selected, and if so, the sub-process proceeds to step 758 in which the selected destination group name is deleted from a member list. From step 758, the sub-process proceeds to step 760 in which a member list of the selected destination group is displayed, the list including at least one group number defined by a corresponding destination IP address and port number. In the preferred embodiment, the member list is displayed in the member list box 734 (FIG. 19) of the GUI screen for defining groups. From step 760, the sub-process proceeds to step 762 in which an interface, allowing a user to add and remove group members in the selected destination group, is provided. Also in step 762, user input indicative of selected added group members, and selected removed group members is received. In the preferred embodiment, step 762 is implemented using the add and remove button 730 and 732 (FIG. 19) of the GUI screen for defining groups. From step 762, the sub-process proceeds to step 764 in which the member list for the selected destination group is updated. In the preferred embodiment, step 764 is implemented by revising the member list that is displayed in the member list box 734 (FIG. 19). From step 764, the sub-process proceeds to "B41'" (back to FIG. 18).

FIG. 21 shows a flow diagram illustrating a sub-process 780 of recording a new asset. The depicted sub-process proceeds from "B5" (from FIG. 13B) to step 782 in which an interface allowing users to select a record path, associated a "record-to" storage unit, is provided, the path indicating a memory storage location for storing the new asset, that is the asset to be encoded and recorded. Also in step 782, user input indicative of a selected record path is received. In the preferred embodiment, the sub-process 780 is implemented using the encoding GUI screen 420 (FIG. 12). Specifically, step 782 is implemented using the "record-to" text box and drop down list 472 of the record asset group box 471 (FIG. 12). From step 782, the sub-process proceeds to step 784 in which an interface prompting the user to enter an asset name for new asset is

provided. Also in step 784, user input indicative of a selected asset name is received. In the preferred embodiment, step 784 is implemented using the asset name text box 476 (FIG. 12) of the encoding GUI screen.

From step 784, the sub-process proceeds to step 786 in which an interface prompting the user to enter a description of content to be encoded is provided. Also in step 786 user input indicative of such a description is received. In the preferred embodiment, step 786 is implemented using the description text box 478 (FIG. 12) which provides for the user to enter a description of the asset to be encoded. From step 786, the sub-process proceeds to step 788 in which an interface prompting the user to select from overwrite options for specifying what to do in the event that the asset name already exists at the record to path, is provided. The override options include a prompt option, an override option, and an append option. Also in step 788, user input indicative of a selected overwrite option is received. In the preferred embodiment, step 788 is implemented using the prompt, overwrite, and append buttons 480, 482, and 484 (FIG. 12) of the encoding GUI screen 420 (FIG. 12).

From step 788, the sub-process proceeds to 790 at which it is determined whether the manual start/stop option was previously selected, and if so, the process proceeds back to "B" (back to FIG. 3). In the preferred embodiment, the manual start and stop option, which is selected using the manual option 432 (FIG. 12), precludes the process from executing the remaining steps 792 and 794 which provide for playing back the newly encoded asset as further explained below.

If it is determined at 790 that the manual option has not been selected, the sub-process proceeds to step 792 in which: an interface, allowing the user to select a schedule playback option, is provided; and user input indicative of the schedule playback option is received. From step 792, the sub-process proceeds to 794 at which it is determined whether the schedule playback option has been selected, and if so, the process proceeds to "B6" (to FIG. 23A).

Alternatively, if it is determined at 794 that the scheduled playback option has not been selected, the process proceeds to "B5(" (back to FIG. 13B).

FIG. 22 shows a block diagram illustrating a playback scheduling GUI screen at 800 for scheduling a playback, or streaming, operation for a newly encoded asset. The GUI screen 800 is activated and displayed upon selection of the schedule playback option button 486 (FIG. 12) of the encoding GUI screen. The GUI screen 800 provides for scheduling an asset to be played back at a later time after encoding of the asset as described above with reference to the encoding GUI screen 420 (FIG. 12). The GUI screen 800 includes: an asset name text box 802 which displays the asset name which is also displayed in the asset name text box 476 (FIG. 12) of the encoding GUI screen, the text box 802 being non-editable; a description text box 804 which is also non-editable, and which displays a description of the asset to be recorded, the description matching the one displayed in the description text box 478 (FIG. 12) of the GUI screen; and a schedule list box 806 for displaying a list of schedule entries, each including graphical information indicative of an asset name for an associated asset, a start date and start time for beginning a play-back schedule for the associated asset, a duration and interval of the play-back schedule, and destination information indicative of a destination or group of destination addresses for the corresponding play-back operation.

The GUI screen 800 also includes a define schedule box 808 including: a start group box 810 providing interface components allowing for the user to define a start date and time for beginning a play-back schedule as further explained below; a destination address group box 812 providing interface components allowing for the user to specify one or more destinations to which the asset is to be streamed in accordance with the play-back operation; a duration group box 814 providing interface components allowing for the user to specify a duration and interval of the play-back schedule as further explained below; an enable notification check box 816 enabling a notification button 818 which is used to activate a GUI screen for defining notification options for notifying selected end users, and selected groups of end users, regarding specified

events under selected circumstances; an add button 820 for adding schedule entries to the schedule list 806; a cancel button 826; an apply button 828; a reset button 830; and a help button 832.

The start group box 810 of the define schedule box 808 includes: a start date drop down list 840 providing for the user to select a start date for the playback schedule, and a start time spin button 842 for selecting a starting time for the playback schedule. The destination group box 812 includes: a manual option button 844 which enables an IP address window 846 for entering a destination IP address, and a port number spin button 848 for selecting a destination port number; a predefined option button 850 which enables a predefined drop down list 852 for selecting a predefined group of destination IP addresses and port numbers, and a define button 854 further explained below; and an SDR button 856.

The duration group box 814 includes: a run option button 860 which enables a run time spin button 862 allowing for the user to specify a play-back scheduled defined by a loop count value as further explained below; a repeat option button 864 allowing the user to define the duration of the playback schedule such that the asset is played repeatedly until the schedule entry associated with the asset is removed from the schedule list 806; and an interval option button 866 which enables a specify button 868 for activating another GUI screen for the purpose of specifying an interval playback schedule as further explained below; and a time to live spin button 870 for specifying a number of routers that the stream can pass through.

FIG. 23A shows a flow diagram illustrating a sub-process at 880 for scheduling playback of a new asset. The depicted sub-process proceeds from "B6" (from FIG. 21) to step 882 in which an asset name and description associated with a currently selected asset is displayed. In the preferred embodiment, the sub-process 880 is implemented using the GUI screen 800 (FIG. 22). From step 882, the sub-process proceeds to step 884 in which a schedule list is displayed. The schedule list includes: a list of assets to be played back, each asset having an associated playback start date and start time, an associated playback duration program, and an associated

playback destination. In the preferred embodiment, the schedule list is implemented by the schedule list box 806 of the scheduling GUI screen 800 (FIG. 22). From step 884, the sub-process proceeds to step 886 in which an interface, allowing the user to select a playback start date and start time for the selected asset, is provided. Also in step 886, user input indicative of the selected start date and start time is received. In the preferred embodiment, step 886 is implemented using the start schedule group box 810 (FIG. 22) of the play-back scheduling GUI screen. From step 886, the sub-process proceeds to step 888 in which an interface, allowing the user to select from playback destination specification options for the selected asset, is provided. The destination specification options include: a manual option, a predefined destination option, a define destination option, and a session director requester (SDR) option. Also in step 888, user input indicative of the selected destination specification option is received. In the preferred embodiment, step 888 is implemented using the destination specification group box 812 (FIG. 22) of the play-back scheduling GUI screen.

From step 888, the sub-process proceeds to 890 at which it is determined whether the manual destination specification option has been selected, and if so, the sub-process proceeds to step 892 in which an interface, allowing the user to select a destination IP address and destination port number, is provided. Also in step 892, user input indicative of the selected destination IP address and selected port number is received. In the preferred embodiment, step 892 is implemented using the IP address text box 846 and the port number spin button 848 (FIG. 22) of the play-back scheduling GUI screen.

Alternatively, if it is determined at 890 that the manual destination specification option has not been selected, the sub-process proceeds from 890 to 894 at which it is determined whether the predefined destination specification option has been selected, and if so, the sub-process proceeds to step 896 in which an interface, allowing the user to select from predefined groups of destination IP addresses and port numbers, is provided. Also in step 896,

user input indicative of the selected group of destination IP addresses and port numbers is received. From step 896, the process proceeds to "B7" (to FIG. 23B).

FIG. 23B shows a flow diagram illustrating further steps of the sub-process 880 (FIG. 23A) for scheduling playback of an asset. The depicted sub-process proceeds from "B7" (from FIG. 23A) to 902 at which it is determined whether the define option has been selected, and if so, the sub-process proceeds to step 904. In step 904, an interface allowing the user to enter a destination group name for a set of IP addresses and port numbers, is provided. Also in step 904, user input including the selected destination group name is received. In the preferred embodiment, step 904 is implemented using the define groups GUI screen 720 (FIG. 19). From step 904, the sub-process proceeds to step 906 in which an interface, allowing the user to select from a list of previously defined destination group names for the purpose of deleting the selected group name is provided. Also in step 906, user input indicative of the selected destination group name is received. In the preferred embodiment, step 906 is implemented using the remove button 732 of the define groups GUI screen 720 (FIG. 19). From step 906, the sub-process proceeds to 908 at which it is determined whether a selected group name is to be deleted, and is so, the sub-process proceeds to step 910 in which the selected destination group name is deleted.

In step 912, a member list for the selected destination group is displayed. The list includes at least one group member defined by a corresponding destination IP address and port number. In the preferred embodiment, the member list is implemented using the member list box 734 (FIG. 19). From step 912, the sub-process proceeds to step 914 in which an interface, allowing the user to add and remove group members in the selected destination group, is provided. Also in step 914, user input indicative of the selected added group members and selected removed group members is received. In step 916, the member list for the selected destination group is updated. From step 916, and also from 902 if it is determined that the define option has not been selected, the sub-process proceeds to step 918 at which it is determined whether the SDR option has been selected. If it is determined at 918 that the SDR option has

been selected, the sub-process proceeds to step 920 to execute the SDR program, after which the process proceeds to "B9" (to FIG. 23C).

FIG. 23C shows a flow diagram illustrating a sub-process at 930 for defining the duration of a play-back schedule for a play-back of an asset. The depicted proceeds from "B9" (from FIG. 23B) to step 932 in which an interface, allowing the user to select from playback schedule duration specification options, is provided. The playback schedule duration specification options include a run option, a repeat option, and a playback interval option. Also in step 932, user input indicative of the selected duration option is received. In the preferred embodiment, step 932 is implemented using the duration box 814 (FIG. 22) of the scheduling GUI screen 800 which includes the run option button 860, the repeat until removed option button 864, and the interval option button 866. From step 932 the sub-process proceeds to step 934 at which it is determined whether the run option has been selected, and if so, the sub-process proceeds to step 936. In step 936, an interface allowing the user to enter a loop count value is provided. The loop count value indicates a number of times that the asset is to be played back. Also in step 936, user input indicative of the selected loop count value is received. In the preferred embodiment, step 936 is implemented using the spin button 862 (FIG. 22) of the scheduling GUI screen 800. In step 938, a duration program is defined based on the selected loop count value selected in step 936.

If it is determined at 934 that the run option has not been selected, the sub-process proceeds from 934 to 940 at which it is determined whether the repeat option has been selected. If it is determined at 940 that the repeat option has been selected, the sub-process proceeds to step 942 in which a playback duration program associated with the selected asset is defined to repeat until removed from the schedule list 806 (FIG. 22).

If it is determined at 940 that the repeat option has not been selected, the sub-process proceeds to 944 at which it is determined whether the interval option has been selected, and if so, the process proceeds to "B10" (to FIG. 25) to execute a sub-process of defining a playback

interval for playing back the to be encoded asset. Alternatively, if it is determined at 944 that the interval option has not been selected, the sub-process proceeds to step 946.

In step 946, an interface allowing the user to select a notification option is provided. Also in step 946, user input indicative of the selected notification option is received. In the preferred embodiment, step 946 is implemented using the enable notification check box 816 (FIG. 22). From step 946, the sub-process proceeds to step 948 at which it is determined whether the notification option has been selected, and if so, the process proceeds to "B11" (to FIG. 27). Alternatively, if it is determined at 948 the notification option has not been selected, the process proceeds to "B14" (to FIG. 23D).

FIG. 23D shows a flow diagram illustrating a sub-process at 950 illustrating further steps of the sub-process 880 (FIG. 23A) for scheduling playback of a new asset. The depicted proceeds from "B14" (from FIG. 23C) to step 952 in which a set of play-back commands and associated parameters is generated for each defined play-back action. Each set of play-back commands and parameters is generated based on: play-back schedule information including the play-back start date and start time defined for the associated play-back action, and the play-back duration program defined for the associated play-back action; and the associated play back-destinations defined for the associated play-back action. From step 952, the sub-process proceeds to step 953 in which the commands and associated parameters associated with each of the play-back actions are transmitted from the administrator terminal 12 (FIG. 1) to the corresponding selected play-back media server for executing a corresponding play-back operation in accordance with the play-back schedule information.

For example, in the preferred embodiment, a set of play-back commands and parameters is transmitted from the administrator terminal to a selected play-back media server associated with the "record-to" location selected using the "record-to" drop down list and window 472 (FIG. 12). The set of play-back commands and parameters programs the selected encoder to execute play-back operations beginning on the play-back start date and start time specified using

the start box 810 (FIG. 22), in accordance with a play-back schedule specified by the user via the GUI components of the duration box 814 (FIG. 22). Each iteration of the play-back operation includes: reading the new asset from the "record-to" location specified using the "record-to" drop down list and window 472 (FIG. 12); and streaming the asset to the play-back destination locations specified by the user via the destination specification box 812 (FIG. 22) of the play-back scheduling GUI screen.

From step 953, the sub-process proceeds to step 954 in which it is again determined whether the notification option has been selected, and if not, the process proceeds to "B5"' (to FIG. 27). In the preferred embodiment, step 954 is implemented using the enable notification check box 816 (FIG. 22). Alternatively, if it is determined at 954 that the notification option has been selected, the sub-process proceeds to step 955 in which a set of notification commands and associated notification parameters is generated for each defined notification action. Each set of notification commands and notification parameters is generated based on user input received in response to GUI screens for defining selected circumstances under which selected notification messages are to be provided to selected end users as further described below with respect to figs. 26 though 30. From step 955, the sub-process proceeds to step 956 in which the notification commands and associated notification parameters associated with each of the play-back actions are transmitted from the administrator terminal 12 (FIG. 1) to the corresponding selected play-back media server for executing a corresponding notification operation.

FIG. 24 shows a block diagram illustrating a GUI screen 960 for defining an interval for a playback schedule. The GUI screen 960 is displayed, and activated, upon activation of the interval option button 866 (FIG. 22). The GUI screen 960 includes: an interval drop down list 962 for selecting an interval from intervals including hourly, daily, weekly, monthly (dates), monthly (days), and yearly; a frequency drop down list 964 allowing the user to select a value associated with the selected interval (i.e., if the selected interval is weekly, the frequency drop down list allows the user to select one of the seven days of the week); a duration box 966 for

defining a time at which the defined interval is to end as further explained below; a current selection text box for displaying the current selection at 968; an OK button 970; a cancel button 972; and a help button 974. The duration box 966 includes: a repeat option button 976 for specifying that the interval is to repeat until removed from the schedule list; and an end date option button 978 which enables a drop down date picker 979 for allowing the user to specify an end date. In the preferred embodiment, the drop down date picker 979 activates a selectable calendar GUI screen as further explained below.

FIG. 25 shows a flow diagram illustrating a sub-process at 980 for defining a playback schedule interval. The sub-process 980 proceeds from "B10" (from FIG. 23C) to step 982 in which an interface, allowing the user to select from interval frequency options, is provided. Also in step 982, user input indicative of the selected frequency option is received. In the preferred embodiment, sub-process 980 is implemented using the define interval GUI screen 960 (FIG. 24). Specifically, step 982 is implemented using the interval drop down list 962 (FIG. 24) and the frequency drop down list 964 (FIG. 24) of the define interval GUI screen. From step 982, the sub-process proceeds to step 984 in which a playback interval is defined based on the interval frequency options selected in step 982. For example, the user may select the weekly interval in the interval drop down list 962 (FIG. 24), and the "every Sunday" frequency using the drop down list 964 (FIG. 24).

From step 984, the sub-process proceeds to step 986 at which an interface, allowing the users to select from duration options, is provided. Also in step 986, user input indicative of the selected duration option is received. In the preferred embodiment, step 986 is implemented using the duration box 966 (FIG. 24) of the defined interval GUI screen 960. From step 986, the sub-process proceeds to step 988 at which it is determined whether the repeat option has been selected. In the preferred embodiment, the repeat option is selected by activating the repeat option button 976 (FIG. 24). If it is determined at 988 that the repeat option has been selected, the sub-process proceeds to step 990 in which an asset is scheduled to be played back in

accordance with the selected frequency option, selected in step 982, until removed from the schedule list.

Alternatively, if it is determined at 988 that the repeat option has not been selected, the sub-process proceeds to step 992 in which it is determined whether the end date option has been selected, and if so, the sub-process proceeds to step 994. In step 994, an interface, allowing the user to select an end date for the scheduled playback, is provided. Also in step 994, user input indicative of the selected end date is received. In the preferred embodiment, the end date is selected by the user using the drop down list 979 (FIG. 24).

From step 994, the sub-process proceeds to step 996 in which an asset to be played back in accordance with the selected frequency option until the selected end date. From step 996, the process proceeds back to "B10'" (back to FIG. 23C).

FIG. 26 shows a block diagram illustrating a GUI screen 1000 for defining notification e-mail messages to be sent to selected persons, and users, in predefined circumstances. The GUI screen 1000 is activated, displayed, upon activating the notification button 818 (FIG. 22) of the asset scheduling GUI screen. The GUI screen 1000 includes: a notification box 1002 having user interface means for defining the circumstances under which e-mail notification is to be sent, specifying destination end users to whom the e-mail messages are to be sent, and defining and editing the contents of the messages to be sent; a notification list window 1004 which displays a list of persons, or end users, to who are to be notified; an add button 1006 for adding notification entries defined using the notification box 1002 to the notification list 1004; a remove button 1008 for removing selected notification entries from the notification list 1004; an OK button 1010; a cancel button 1012; and a apply button 1014; a reset button 1016; and a help button 1018.

The notification box 1002 includes: a notification circumstances option window 1019 having a plurality of option buttons for defining circumstances under which e-mail notification is to be provided to selected persons, the notification option buttons including a delivery fails

option button 1020 for specifying an e-mail is to be sent to selected persons when streaming of the asset fails, a deliver succeeds button 1022 specifying that notification are to be sent to the selected persons upon success of (??), and an always notification option button 1024 specifying that e-mail messages are to be sent to the selected persons in the event of failure or success; a drop down list 1026 allowing the user to select who they want an e-mail notification sent to (groups?); a define button 1028 for activating a define groups GUI screen for defining groups of e-mail addresses as further explained below; a current message text box 1030 which is a non-editable for displaying the e-mail message to be sent to the selected destination addresses; and an edit message button 1032 for activating an edit message GUI screen for editing the current message to be sent as further explained below.

FIG. 27 shows a flow diagram illustrating a sub-process at 1050 of providing a user interface allowing a user to define circumstances under which notification e-mail messages are to be sent to selected persons, user interface also allowing the user to draft and edit the e-mail messages and select the destination e-mail addresses to which the messages are to be sent. In the preferred embodiment, the sub-process 1050 is implemented using the GUI screen 1000 (FIG. 26). The depicted sub-process proceeds from "B11" (from FIG. 23C) to step 1052 in which a notification list including previously defined notification entries is displayed. In the preferred embodiment, the notification list is implemented using the notification list box 1004 (FIG. 26) of the define notification GUI screen. From step 1052, the sub-process proceeds to step 1054 in which an interface, allowing the user to select from notification circumstance options, is provided. Also in step 1054 user input indicative of a selected notification circumstance option is received. In the preferred embodiment, step 1054 is implemented using the notification circumstance option window 1019 (FIG. 26) which provides for the user to select from the options of sending e-mail notification messages when delivery fails, when delivery succeeds, and when delivery fails or succeeds, that is always. From step 1054, the sub-process proceeds to step 1056 in which an interface, allowing the user to select from groups of destination e-mail

addresses, is provided. Also in step 1056, user input indicative of a selected group is received. In the preferred embodiment, step 1056 is implemented using the drop down list 1026 (FIG. 26) which provides a list of predefined groups of destination e-mail addresses.

From step 1056, the sub-process proceeds to step 1060 in which an interface, allowing the user to select an option of defining a new group of destination e-mail addresses, is provided. Also in step 1060, user input indicative of the selected option is received. From step 1060, the sub-process proceeds to 1062 at which it is determined whether the option of defining a new group has been selected, and if so, the process proceeds to "B12" (to FIG. 29) to execute a sub-process of defining a group of destination e-mail addresses as further explained below.

Alternatively, if it is determined at 1062 that the option of defining a new group has not been selected, the sub-process proceeds to step 1064 in which a current message to be sent to the selected group is displayed. In the preferred embodiment, the current message is displayed in the current message window 1030 (FIG. 26). From step 1064, the sub-process proceeds to step 1066 in which an interface, allowing the user to select the option of editing the current message, is provided. Also in step 1066 user input indicative of the user's response to the option is received. In the preferred embodiment, the option of editing the messages provided by the edit message box 1032 (FIG. 26). From step 1066, the sub-process proceeds to step 1068 at which it is determined whether the option of editing the message has been selected, and if so, the process proceeds to "B13" (to FIG. 31) to execute a sub-process of providing a user interface for editing the message to be sent. Note that the current message box 1030 is not editable.

In step 1070, a new notification entry is defined based on the user selected notification circumstances, selected notification destination e-mail addresses, and the contents of the current message. Also in step 1070, the new notification entry is added to the notification list. From step 1070, the sub-process proceeds to step 1072 in which the revised notification list is displayed. In the preferred embodiment, the revised notification list is displayed in the

notification list box 1004 (FIG. 26). From step 1072, the process proceeds back to "B11'" (back to FIG. 23C).

FIG. 28 shows a block diagram illustrating a GUI screen 1080 for defining groups of destination e-mail addresses for the purpose of sending notification messages to the destination addresses. The GUI screen 1080 is displayed and activated upon activation of the define button 1028 (FIG. 26) of the define notification GUI screen. The GUI screen 1080 includes: a group name text box 1082 providing for the user to enter a name for a set of e-mail addresses; a delete button 1084 which is enable if there are previously defined groups listed in the drop down list 1082, the delete button providing for deleting a selected group name including the e-mail addresses associated with the selected group name; a define group box 1086 providing user interface means used for defining a set of e-mail addresses associated with the group name displayed in the group name text box 1082.

The defined group box 1086 includes: an e-mail address text box 1090 providing for the user to enter a destination e-mail address; a member list box 1092 for displaying a list of e-mail addresses associated with the group name displayed in the group name text box 1082, the member list box being multi-selectable; an add button 1094 for adding the destination e-mail address listed in the address box 1090 to the member list displayed in the member list box 1092; a remove button 1096 for removing selected ones of the e-mail addresses listed in the member list from the member list; an OK button 1098; a cancel button 1100; and a help button 1102.

FIG. 29 shows a flow diagram illustrating a sub-process at 1110 for defining a group of destination e-mail addresses for the purpose of sending notification messages to the group. In the preferred embodiment, the sub-process 1110 is implemented using the define groups GUI screen 1080 (FIG. 28). The sub-process 1110 proceeds from "B12" (from FIG. 27) to step 1112 in which an interface, allowing the user to enter a group name for a new group of e-mail addresses, is provided. Also in step 1112, user input indicative of a group name is received. In the preferred embodiment, step 1112 is implemented using the group name text box and drop down

list 1082 (FIG. 28). From step 1112, the sub-process proceeds to step 1114 in which the member list including e-mail addresses associated with the selected group name is displayed. In the preferred embodiment, the member list is displayed in the member list box 1092 (FIG. 28). From step 1114, the sub-process proceeds to step 1116 in which an interface, allowing the user to select add and remove options for adding and removing members from the current member list, is provided. Also in step 1116, user input indicative of the selected add/remove option is received. In the preferred embodiment, step 1116 is implemented using the add and remove buttons 1094 and 1096 (FIG. 28) of the define groups GUI screen. From step 1116, the sub-process proceeds to 1118 at which it is determined whether the remove option has been selected, and if so, the sub-process proceeds to step 1120 in which a user interface, allowing the user to select a member from the member list, is provided. Also in step 1120, user input indicative of the selected member is received. In the preferred embodiment, members listed in the member list box 1092 (FIG. 28) are multi-selectable (shift-contiguous; CTRL-disjoint selection). Also in step 1120, user input indicative of a selected member is received.

From step 1120, the sub-process proceeds to step 1122 in which the selected member is removed from the list.

If it is determined at 1118 that the remove option has not been selected, it is assumed that the add option has been selected, and the sub-process proceeds to step 1126 in which a user interface, allowing the user to enter an e-mail address associated with a new member is provided. Also in step 1126 user input indicative of an e-mail address associated with a new member is received. From step 1126, the sub-process proceeds to step 1128 in which the e-mail associated with the new member is added to the member list displayed in the member list box 1092 (FIG. 28). From step 1128, the sub-process proceeds to step 1130 in which the revised member list is displayed. In the preferred embodiment, the revised member list is displayed in the member list box 1092 (FIG. 28) of the define groups GUI screen. From step 1130, the process proceeds back to "B12"' (back to FIG. 27).

FIG. 30 shows a block diagram illustrating an edit message GUI screen 1140. The screen 1140 is activated in response to the user activating the edit message button 1032 (FIG. 26) of the defined notification GUI screen. The edit message GUI screen 1140 includes: header attributes including an asset name check box 1142 for including an asset name in the message; a status attribute check box 1144 for including the status of the message; a start check box 1146 for specifying that the start date and time of a multicast is to be included in the e-mail message; a error code check box 1150 for specifying a predefined error code indicative of a failure in delivery of a multicast; and a destination box 1152 for specifying the destination e-mail address of or associated with the message. The GUI screen 1140 further includes a current message text box 1154 for displaying a current message, the contents of the current message text box being editable by the user; an OK button 1156; a cancel button 1158; and a help button 1160.

FIG. 31 shows a flow diagram illustrating a sub-process at 1170 for editing a message to be sent to selected e-mail destination addresses. The depicted sub-process proceeds from "B13" (from FIG. 27) to step 1172 in which an interface, allowing the user to select from a plurality of header attributes, is provided. The selected attributes are to be added to the e-mail message. Also in step 1172, user input indicative of selected header attributes are received. In the preferred embodiment, step 1172 is implemented using the header attributes box 1141 of the edit message GUI screen 1140 (FIG. 30).

From step 1172, the sub-process proceeds to step 1174 in which the current message is displayed. In step 1176, an interface allowing the user to edit the current message is provided. Also in step 1176 user input for editing the current is received. In step 1178, the revised current message is displayed. In the preferred embodiment, the steps 1174, 1176, and 1178 are implemented using the current message text box 1154 of the edit message GUI screen 1140 (FIG. 30). From step 1178, the process proceeds to "B13'" (back to FIG. 27).

FIG. 32 shows a block diagram illustrating a GUI screen 1200 which provides an interface for multicasting existing assets. The multicasting GUI screen 1200 is activated upon

selection of the multicast option. The GUI screen 1200 includes an asset select box 1202 for selecting an existing asset located at a selected one of the available media servers (i.e., SERVER_1 or SERVER_2 of FIG. 1), the box 1202 including: an available server drop down list 1204 providing for the user to select a media server that contains assets; a server time text box 1206 for displaying the local time at the selected server, the server time being active (the seconds are clicking away in real time); an existing assets list box 1208 for displaying a list of asset names associated with assets located at the selected server, and also displaying the date and time that the corresponding asset was last modified; and a description label box 1210 for displaying graphical information indicative of a description of the selected asset located at the selected server.

The GUI screen 1200 further includes: a define schedule box 1212 providing user interface means for defining a schedule including a start date and time, a duration, and destination end users associated with the selected asset at the selected media server and also for defining notification messages to be sent to selected persons under selected circumstances as further explained below; a schedule list 1214 for displaying a list of schedule entries each of the schedule entries including an asset name, a start date and time for multicasting the asset, a duration for the scheduled program associated with the asset, and destination locations to which the asset is to be streamed; a remove button 1216 for removing selected ones of the scheduled entries from the schedule list 1214; an OK button 1218; a cancel button 1220; an apply button 1222; a reset button 1224; and a help button 1226.

The define schedule box 1212 is substantially similar to the define schedule box 808 of the playback scheduling GUI screen 800 (FIG. 22), and includes a start box 1230. The start box 1230 includes: a start now option button 1232 for specifying that the selected asset is to be multicasted from the selected server immediately; and a schedule time option button 1234 for specifying a start date and start time for a program for multicasting the selected asset from the selected server. The schedule time option button 1234 enables a start date drop down date

picker 1236 allowing the user to select a date via a selectable calendar view as further explained below. and a start time spin button 1238 allowing for the user to select a start time for beginning the mutlicasting schedule.

The GUI screen 1200 further includes: a destination group box 1240 which is identical to, and which includes all of the same interface components as, the destination group box 812 of the playback scheduling GUI screen 800 (FIG. 22); and a duration group box 1240 which is identical to, and which includes all of the same interface components as, the duration group box 814 of the playback scheduling GUI screen 800 (FIG. 22).

An enable notification check box 1244 enables a notification button 1246 which is used to activate a GUI screen for defining notification options for notifying selected end users, and selected groups of end users, regarding specified events under selected circumstances. Selecting the notification button 1246 causes the GUI process to activate the GUI screen 1000 (FIG. 26) for defining notification e-mail messages to be sent to selected persons, and users, in predefined circumstances. A define button 1248 provides for adding schedule entries to the schedule list 1214.

FIG. 33A shows a flow diagram illustrating a multicasting sub-process at 1250 in accordance with the present invention. The sub-process 1250 proceeds from "C" (from FIG. 4) to step 1252 in which an interface, allowing the user to select from a displayed list of media servers is provided. Also in step 1252, user input indicative of a selected one of a media servers is received. In the preferred embodiment, the sub-process 1250 is implemented using the multicasting GUI screen 1200 (FIG. 32). From step 1252, the sub-process proceeds to step 1254 in which the local time at the selected media server is displayed. In the preferred embodiment, the local time is displayed in the server time window 1206 (FIG. 32) of the multicasting GUI screen.

From step 1254, the sub-process proceeds to step 1256 in which a list of assets located at the selected media server is displayed. Also in step 1256, a corresponding date and time that

each of the assets were last modified is displayed. In the preferred embodiment, step 1256 is implemented using the existing assets display box 1208 (FIG. 32) of the multicasting GUI screen. From step 1256, the sub-process proceeds to step 1258 in which a user interface, allowing the user to select from the list of assets located at the selected location, is provided. Also in step 1258, user input indicative of the selected asset is received. From step 1258, the sub-process proceeds to step 1260 in which a description of the selected asset is displayed. In the preferred embodiment, the description of the selected asset is displayed in the description box 1210 (FIG. 32) of the multicasting GUI screen. From step 1260, the sub-process proceeds to step 1262 in which a schedule list, including a list of assets to be played back, is displayed. Each of the listed assets has an associated start date and time, and associates playback duration program, and an associated playback destination, all of which are displayed along with the schedule list. In the preferred embodiment, step 1262 is implemented using the schedule list display box 1214 (FIG. 32) of the multicasting GUI screen. In step 1264, a user interface, allowing the user to select from multicast start options, is provided.

The multicasting start options include an immediate start option, and a scheduled start option. Also in step 1264, user input indicative of the selected multicast start option is received. In the preferred embodiment, step 1264 is implemented using the start box 1230 (FIG. 32) of the multicasting GUI screen.

From step 1264, the sub-process proceeds to step 1266 at which it is determined whether the immediate start option has been selected. The determination at 1266 is made and the preferred embodiment by determining the "now" option button 1232 (FIG. 32) has been selected. If it is determined at 1266 that the immediate start option has not been selected, the sub-process proceeds to step 1268 at which an interface, allowing the user to select a multicast start date and start time for the selected asset, is provided. Also in step 1268, user input indicative of the selected start date and start time is received. In the preferred embodiment, step 1268 is implemented using the start date drop down list 1236 and the start time spin button 1238 (FIG.

32) of the multicasting GUI screen. Alternatively, if it is determined at 1266 that the immediate start option has been selected, the sub-process proceeds from 1266 to step 1270 in which a multicast start time is defined based on the selected multicast start option. If the immediate start option has been selected, the multicast start time is defined to be a time immediately upon selection of the OK button 1218 (FIG. 32). If the scheduled start option has been selected, the start time is defined to be the selected start date and selected start time. From step 1270, the process proceeds to "C1" (to FIG. 33B).

FIG. 33B shows a flow diagram illustrating a sub-process 1280 for specifying destinations to which the existing asset is to be multicasted, or streamed. In the preferred embodiment, the sub-process 1280 is implemented using the destination box 1240 (FIG. 32) of the multicasting GUI screen. The sub-process 1280 proceeds from "C1" (from FIG. 33A) to step 1282 in which an interface, allowing the user to select from multicast destination specification options, is provided. The multicast destination specification options include a manual specification option, a predefined option, a defined option, and an SDR option. Also in step 1282, user input indicative of the selected option is received. From step 1282, the sub-process proceeds to step 1284 at which it is determined whether the manual option has been selected. In the preferred embodiment, the manual option is determined to be selected if the manual option button 844 (FIG. 32) has been selected by the user. If it is determined at 1284 that the manual option has been selected, the sub-process proceeds to step 1286 in which an interface, allowing the user to select a destination IP address and destination port number, is provided. Also in step 1286, user input indicative of the selected destination IP address and selected port number is received. In the preferred embodiment, step 1286 is implemented using the IP address text box 846 and the port number spin button 1246 (FIG. 32). Alternatively, if it is determined at 1284 that the manual option has not been selected, the sub-process proceeds to 1288 at which it is determined whether the predefined option has been selected. In the preferred embodiment, the

predefined option is determined to be selected if the predefined option button 850 (FIG. 32) has been selected by the user.

If it is determined at 1288 that the predefined option button has been selected, the sub-process 1280 proceeds to step 1290 in which an interface, allowing the user to select from predefined of group of destination specification IP addresses and port numbers, is provided. Also in step 1290, user input indicating a selected group of destination IP addresses and port numbers is received. In the preferred embodiment, step 1290 is implemented using the predefined group drop down list 852 (FIG. 32) of the multicasting GUI screen. From step 1290, the process proceeds to "C2" (to FIG. 33C).

FIG. 33C shows a flow diagram illustrating further steps of the destination specification and multicasting sub-processes. The sub-process 1300 proceeds from "C2" (from FIG. 33B) to 1302 at which it is determined whether the define destination specification option has been selected. In the preferred embodiment, the define option is determined to be selected if the define button 854 (FIG. 32) has been selected by the user.

If it is determined at 1302 that the define option has been selected, the sub-process proceeds to step 1304 in which an interface, allowing the user to enter a destination group name for a corresponding set of IP addresses and port numbers is provided. Also in step 1304, user input indicative of a selected destination group name is received. In the preferred embodiment, step 1304 is implemented using the define groups GUI screen 720 (FIG. 19).

From step 1304, the sub-process proceeds to step 1306 in which an interface, allowing the user to select from a list of previously defined destination group names for the purpose of deleting the selected group name, is provided. Also in step 1306, user input indicative of the selected destination group name is received. In the preferred embodiment, step 1306 is implemented using the member list box 734 (FIG. 19) and the remove button 732 (FIG. 19) of the define group GUI screen. From step 1306, the depicted sub-process proceeds to step 1308 at

which it is determined whether a selected group name is to be deleted, and if so, the sub-process proceeds to step 1310 in which the selected destination group name is deleted.

Alternatively, if it is determined at 1308 that a selected destination group name is not to be deleted, the sub-process proceeds to step 1312 in which a member list is displayed for the selected destination group. The member list includes a least one group member defined by a corresponding destination IP address and destination port number. In the preferred embodiment, step 1312 is implement using the member list box 734 (FIG. 19) of the defined groups GUI screen. From step 1312, the sub-process proceeds to step 1314 in which an interface, allowing the user to add and remove group members in the selected destination group, is provided. Also in step 1314 user input indicative of the selected added group members and selected remove group members is received. In the preferred embodiment, step 1314 is implemented using the member list box 734 (FIG. 19) which allows for selecting members from the member list, the add button 730 (FIG. 19), and a remove button 732 (FIG. 19).

From step 1314, the sub-process proceeds to step 1316 in which a member list associated with the selected destination group is updated. From step 1316, the sub-process proceeds to step 1318 at which it is determined whether the SDR option has been selected, and if so, the sub-process proceeds to step 1320 to execute an SDR program. Alternatively, if it is determined at 1318 that the SDR option has not been selected, the process proceeds to "C3" (to FIG. 33D).

FIG. 33D shows a flow diagram illustrating a sub-process at 1330 for defining a duration for a schedule multicasting existing assets. The sub-process 1330 proceeds from "C3" (from FIG. 33C) to step 1332 in which an interface, allowing a user to select playback duration program options, is provided. The playback duration program options include a run option, a repeat option, and a playback interval option. Also in step 1332, user input indicative of a selected duration option is received. The sub-process 1330 is identical to the sub-process 930 (FIG. 23C).

FIG. 33E shows a flow diagram illustrating a sub-process at 1350 illustrating further steps of the sub-process 1250 (FIG. 33A) for defining and scheduling a multicast operation. The depicted process proceeds from "C12" (from FIG. 33D) to step 1352 in which a set of multi-casting commands and associated parameters is generated for each defined multi-casting action. Each set of multi-casting commands and parameters is generated based on: multi-casting schedule information including the multi-casting start date and start time defined for the associated multi-casting action, and the multi-casting duration program defined for the associated multi-casting action; and the associated multi-casting destinations defined for the associated multi-casting action. From step 1352, the sub-process proceeds to step 1353 in which the commands and associated parameters associated with each of the multi-casting actions are transmitted from the administrator terminal 12 (FIG. 1) to the corresponding selected multi-casting media server for executing a corresponding multi-casting operation in accordance with the multi-casting schedule information.

For example, in the preferred embodiment, a set of multi-casting commands and parameters is transmitted from the administrator terminal to a selected media server. The set of multi-casting commands and parameters programs the selected media server to execute multi-casting operations beginning on the play-back start date and start time specified using the start box 1236 (FIG. 32), in accordance with a multi-casting schedule specified by the user via the GUI components of the duration box 1260 (FIG. 32). Each iteration of the multi-casting operation includes streaming the asset to the multi-casting destination locations specified by the user via the destination specification box 1240 (FIG. 32) of the multi-casting GUI screen.

From step 1353, the sub-process proceeds to 1354 at which it is again determined whether the notification option has been selected, and if not, the process proceeds to "C" (to FIG. 27). In the preferred embodiment, the determination at 1354 is implemented using the enable notification check box 1244 (FIG. 32). Alternatively, if it is determined at 1354 that the notification option has been selected, the sub-process proceeds to step 1355 in which a set of

notification commands and associated notification parameters is generated for each defined notification action. Each set of notification commands and notification parameters is generated based on user input received in response to GUI screens for defining selected circumstances under which selected notification messages are to be provided to selected end users. From step 1355, the sub-process proceeds to step 1356 in which the notification commands and associated notification parameters associated with each of the play-back actions are transmitted from the administrator terminal 12 (FIG. 1) to the corresponding selected media server for executing a corresponding notification operation.

FIG. 34 shows a block diagram of a selectable calendar GUI screen at 1360, the screen 1360 including forward and reverse selection buttons 1362 and 1364 for displaying days and dates of successive months in a calendar view. The user may pick a date by selecting a number representing a date as indicated at 1366. The screen 1360 is displayed and activated upon activation of the start date drop down list 1236 (FIG. 32) of the multicasting GUI screen.

FIG. 35 shows a block diagram illustrating a global schedule summary Web document at 1400 in accordance with the present invention, the Web document 1400 including graphical information indicative of a plurality of events, or operations, which have been scheduled for execution at corresponding media servers by a particular user, or administrator, using the media asset management and scheduling GUI process of the present invention as described above. The Web document 1400 is provided by a global asset management server which receives the server identification information, schedule information, and operation information associated with each media operation scheduled by authorized users, or administrators. In order to provide the Web document 1400, the asset management and scheduling GUI process of the present invention includes further steps for transmitting the server identification information, schedule information, and operation information associated with each user defined media operation from the administrator terminal 12 (FIG. 1) to the global asset management server. The global asset management server stores a global schedule summary file for each authorized user, the summary

file including all server identification information, operation information, and scheduling information associated with each operation, or event, scheduled by the corresponding user via the asset management and scheduling GUI process of the present invention described above.

In one embodiment of the present invention, the Web document 1400 is provided by SERVER_1 (FIG. 1) which executes the asset management and scheduling program 23 (FIG. 1), and which serves as the global asset management server. In this embodiment, SERVER_1 receives the server identification information, schedule information, and operation information associated with each media operation from the administrator terminal 12 (FIG. 1) at the same time that the commands and associated parameters associated with the media operations are transmitted to the media server indicated by the server identification information. The global asset management server provides a centralized location for storage of information associated with all scheduled media operations. The Web document 1400 may be accessed by a user via a Web browser application executed by the administrator terminal 12 (FIG. 1) using a corresponding URL. In a preferred embodiment of the present invention, the Web document 1400 is a HTML document including embedded Java applets which are executed upon activation of corresponding active areas of the HTML document as further explained below.

The Web document 1400 includes: a schedule time interval selection area 1402 providing a group of GUI components which allow the user to select a time interval, a time zone, a date, and a start time for defining a selected time interval of the global schedule summary; and a global schedule display area 1404 for displaying event programming information indicative of media operations scheduled for execution during the selected time interval defined using the GUI components of the time interval selection area 1402 as further explained below.

The time interval selection area 1402 includes: a time interval drop down list 1406 allowing the user to select a time interval; a time zone drop down list 1410 allowing the user to select from a plurality of time zones, the selected time zone providing a reference for the selected time interval; a date drop down list 1412 allowing the user to select a date for the selected time

interval of the global schedule summary associated with the particular user; and a start time drop down list 1414 allowing the user to select a start time for defining the selected time interval of the global schedule summary associated with the particular user.

The program schedule area 1404 includes: a plurality of entries 1418 for displaying graphical indicia indicative of information associated with corresponding media operations which have been defined and schedule by the particular user for execution by a selected media server during the selected time interval; a plurality of time interval columns 1420 delineating time sub-intervals of the selected time interval of the global event schedule summary; and a subject column 1422 for displaying graphical subject indicia indicative of a subject associated with at least one corresponding media operation which has been defined and scheduled by the particular user to be executed by a selected media server during the selected time interval.

For each of the entries 1418, the program schedule area includes corresponding subject indicia 1424 indicative of a subject associated with at least one corresponding media operation, and corresponding operation indicia 1426 indicative of corresponding media operations to be performed by corresponding selected media servers during a corresponding time sub-interval indicated by the particular time interval columns 1420 between which the operation indicia 1426 is located. Each operation indicia 1426 is associated with an active area as further explained below.

For example, an entry of the subject column 1422 may include a subject indicia 1424 indicative of a particular course offered by an educational institution (e.g., Engineering 101), and an operation indicia 1426 indicative of a selected media operation to be performed on an asset comprising audio and video data representing a lecture associated with the particular course. The selected media operation indicated by the operation indicia 1426 may be selected from the above defined media operations including copy operations, delete operations, encoding operations, multicasting operations, and notification operations.

Upon selection of one of the active areas associated with one of the operation indicia 1426, an applet is activated, and interpreted by a Web browser at the administrative terminal 12 (FIG. 1), the applet providing at least a portion of the asset management and scheduling GUI process of the present invention. Clicking on, or activating, one of the active areas associated with one of the operation indicia 1426 causes the GUI process of the present invention to be executed, and initialized to display a corresponding one of the GUI screens described above which is associated with defining and scheduling the media operation associated with the selected operation indicia 1426. The user may then use the GUI process of the present invention to redefine, reschedule, or cancel the corresponding media operation associated with the selected operation indicia 1426 as further explained below.

FIG. 36 shows a flow diagram illustrating a process at 1450 for managing global schedule information in accordance with the present invention. In one embodiment, the process 1450 is implemented as computer readable code executed by the processing unit of SERVER_1 (FIG. 1) which provides a global asset management server. The process 1450 begins with step 1452 in which summary information associated with a particular media operation is received from the administrator terminal 12 (FIG. 1). The summary information includes: user identification information indicating a corresponding user who has defined and scheduled the particular media operation using the asset management and scheduling GUI process described above; server identification information indicating a selected media server scheduled to execute the particular media operation; operation information defining a particular media operation (e.g., defining the particular media operation as a copy asset operation, a delete asset operation, an encoding asset operation, a multicasting operation, or a notification operation); and schedule information specifying a schedule for executing the particular media operation. In an embodiment, the summary information associated with each particular media operation is transmitted from the administrator terminal 12 to SERVER_1 (FIG. 1) at the same time that the commands and associated parameters associated with the particular media operation are transmitted from the

administrator terminal to the selected media server which is indicated by the server identification information.

From step 1452, the process proceeds to step 1454 in which the server identification information, operation information, and schedule information associated with the particular media operation is stored in a global schedule file associated with the indicated user. In an embodiment, the global schedule file associated with the indicated user is stored on the local disk 22 (FIG. 1) of SERVER_1.

FIG. 37 shows a flow diagram illustrating a process at 1470 for generating a global schedule Web document in accordance with the present invention. In the described embodiment, the process 1470 is implemented as computer readable code executed by the processing unit of SERVER_1 (FIG. 1) which acts as the global asset management server. The process 1470 begins with step 1472 in which user information specifying a user, or administrator, is received at SERVER_1 from the administrator terminal 12 (FIG. 1) via the network. From step 1472, the process proceeds to step 1474 in which the global asset management server reads a global schedule file associated with the user specified in step 1472. From step 1474, the process proceeds to step 1476 in which the global asset management server generates a global schedule Web document associated with the specified user based on the global schedule file associated with the user. The global schedule Web document includes: a plurality of operation graphical indicia associated with corresponding operations defined by the specified user; server identification information, operation information, and schedule information associated with each corresponding operation graphical indicia; active areas associated with each corresponding operation graphical indicia; an applet providing the asset management and scheduling GUI process of the present invention; and links to media servers indicated by the server identification information. In one embodiment, the global schedule Web document 1400 (FIG. 35) provides the format for the document generated in step 1476.

FIG. 38 shows a flow diagram illustrating a process at 1500 for reviewing a summary of all scheduled media operations associated with a particular user of the asset management and scheduling GUI process of the present invention, the process 1500 also providing for editing operation information and schedule information associated with selected ones of the scheduled media operations associated with the particular user. In the described embodiment, the process 1500 is implemented as computer readable code executed by the processing unit of the administrator terminal 12 (FIG. 1). The process 1500 begins with step 1501 in which user identification information specifying a user is transmitted from the administrator terminal 12 (FIG. 1) to the global asset management server, that is SERVER_1 (FIG. 1) in the described embodiment. In one embodiment, step 1501 includes accessing SERVER_1 via the Web browser using a predetermined uniform resource locator (URL). From step 1501, the process proceeds to step 1502 in which a global schedule Web document associated with the specified user is received at the administrator terminal 12 (FIG. 1). The global schedule Web document includes: a plurality of operation graphical indicia associated with corresponding operations defined by the specified user; server identification information, operation information, and schedule information associated with each corresponding operation graphical indicia; active areas associated with each corresponding operation graphical indicia; an applet providing the asset management and scheduling GUI process of the present invention; and links to media servers indicated by the server identification information. In the preferred embodiment, the Web document is implemented by the Web document 1400 (FIG. 35).

In step 1504, the Web document is displayed in a browser window of the administrative terminal 12 (FIG. 1). From step 1504, the process proceeds to step 1506 in which user input indicating a selected operation graphical indicia, associated with a selected media operation, is received. In the preferred embodiment, the user selects an operation graphical indicia 1426 (FIG. 35) by selecting (e.g., "clicking on") an active area which is coincident with the selected operation graphical indicia (FIG. 35). In step 1507, an applet embedded in the Web document is

executed at the administrator terminal to provide the asset management and scheduling GUI process of the present invention as described above. In the preferred embodiment, the applet is a Java applet which is embedded in the Web document 1400 (FIG. 35).

In step 1508, operation information and schedule information associated with a selected scheduled event is read. In step 1510, the asset management and scheduling GUI process of the present invention is initiated to display specific GUI screens for editing the operation information and schedule information associated with the selected scheduled event. In step 1510, the asset management and scheduling GUI process advances to the GUI screen associated with the scheduled event. For example, if the scheduled event is a copy action or a delete action, the asset management and scheduling GUI process proceeds to "A" (to FIG. 6). As another example, if the selected scheduled event is a scheduled encoding operation, the process proceeds to "B" (to FIG. 13A). As yet a further example, if the selected scheduled event is a scheduled multicasting operation, the process proceeds to "C" (back to FIG. 33A). In any of these cases, the GUI process displays the appropriate GUI screen having the operation information and schedule information associated with the selected media operation displayed therein for editing.

From step 1510, the process proceeds to step 1512 in which a link is established with the media server indicated by the server identification information associated with the selected media operation. In step 1514, existing commands and associated parameters associated with the selected event are deleted at the source media server associated with the selected event. From step 1514, the process proceeds to "J" (to FIG. 4) to execute the asset management and scheduling GUI process in order to redefine, reschedule, or cancel the selected media operation.

Although the present invention has been particularly shown and described above with reference to a specific embodiment, it is anticipated that alterations and modifications thereof will no doubt become apparent to those skilled in the art. It is therefore intended that the following claims be interpreted as covering all such alterations and modifications as fall within the true spirit and scope of the invention.

The embodiments of the invention in which an exclusive property or privilege is claimed are defined as follows:

1. A process for providing a centralized user interface at an administrator terminal in a network for scheduling and coordinating the execution of operations performed on media data by selected ones of a plurality of media servers of the network, comprising the steps of:

displaying a graphical user interface at the administrator terminal, the interface including a plurality of interface components enabling a user to specify and schedule operations to be performed by selected ones of the media servers;

receiving information input by the user identifying a selected one of the media servers, an operation to be performed by the selected media server, and a schedule for performing the operation;

generating commands and associated parameters based on the input information for instructing the selected media server to execute the specified operation in accordance with the specified schedule; and

transmitting the commands and associated parameters to the selected media server.

2. A process as recited in claim 1 wherein the network is an internet protocol (IP) network.

3. A process as recited in claim 1 wherein the administrator terminal also includes a processing unit, and a browser application executed by the processing unit, and wherein the steps of displaying, receiving, generating, and transmitting are performed by executing an applet over the processing unit.

4. A process as recited in claim 1 wherein the administrator terminal also includes a processing unit, and a browser application executed by the processing unit, and wherein the process is initiated by performing the steps of:

transmitting an applet to the administrator terminal via the network; and

executing the applet over the processing unit of the administrator terminal;

whereby the graphical user interface is displayed within a browser window generated by the browser application.

5. A process as recited in claim 1 wherein at least one of the media servers is further operative to encode media data received from a corresponding multimedia device, and wherein the operations include:

delete operations for deleting portions of media data from catalogs of predetermined mapped ones of the media servers;

copy operations for copying portions of media data from selected source locations in the network to selected destination locations in the network

multicasting operations for streaming portions of media data from selected ones of the media servers to selected ones of a plurality of end user terminals via the network; and

encoding operations for encoding media data received by selected ones of the media servers.

6. A process as recited in claim 1 wherein at least one of the media servers is also operative to encode and parse portions of media data to create media assets, and wherein each of the media servers is further operative to maintain a catalog including a plurality of pointers to catalogued ones of the media assets stored in a local disk included within the media server.

7. A process as recited in claim 1 wherein each of the media servers is operative to access a memory device for storing media data, and also operative to transfer portions of media data to selected locations via the network, and wherein the operations include copy operations for copying selected portions of media data from selected source locations in the network to selected destination locations in the network.

8. A process as recited in claim 7 wherein the steps of displaying a graphical user interface, and receiving input information further comprise the steps of:

displaying a source selection interface component enabling the user to select a source location by browsing a list of available locations including predetermined mapped ones of the media servers and predetermined mapped ones of the memory devices;

receiving information indicative of a selected source location;

displaying a media data selection interface component enabling the user to select a portion of media data by browsing a list of available a portions of media data stored at the selected source location;

receiving information indicative of a selected portion of media data;

determining a corresponding list of possible destination locations associated with the selected source location; and

displaying a destination selection interface component enabling the user to select a destination location by browsing the corresponding list of possible destination locations.

9. A process as recited in claim 8 wherein the steps of displaying a graphical user interface, receiving information, and generating commands further comprise the steps of:

displaying delayed copying interface components enabling the user to select a start time and a start date for a delayed copying operation;

receiving information indicative of a selected start time and a selected start date; and

generating a copy command and associated parameters for instructing the selected media server to copy the selected portion of media data from the selected source location to the selected destination location at the selected start time on the selected start date.

10. A process as recited in claim 7 further comprising the steps of:

displaying a location mapping interface component enabling the user to define a path to a new location in the network;

receiving information indicative of a selected path for accessing the new location; and

adding the new location associated with the selected path to the list of available locations.

11. A process as recited in claim 8 wherein the memory devices include local disks included within corresponding ones of the media servers, and remote disks and archives communicatively coupled with corresponding ones of the media servers, and wherein each of the media servers is further operative to encode and parse portions of media data to create media assets, and wherein each of the media servers is further operative to maintain a catalog including pointers to catalogued ones of the media assets stored the local disk of the media server, and wherein the copy operations performed by the selected media server include:

an adding operation for adding selected portions of media data from the local disk of the selected media server to a catalog of the selected media server, the adding operation including a sub-step of parsing a selected portion of media data if it is not parsed;

a loading operation for loading selected portions of media data from selected remote disks coupled with the selected media server to a catalog of the selected media server, the loading operation including a sub-step of parsing a selected portion of media data if it is not parsed; and

staging assets from the selected media server to a destination location selected from the archives and the media servers.

12. A process as recited in claim 10 wherein the step of determining a corresponding list of possible destination locations comprises:

determining whether the selected source location is a media server, and if the selected source location is a media server, defining the list of possible destination locations to include predetermined mapped ones of the media servers other than the selected source media server, and predetermined mapped ones of the archives;

determining whether the selected source location is an archive, and if the selected source location is an archive, defining the corresponding list of possible destination locations to include

predetermined mapped ones of the archives other than the selected source archive, and predetermined mapped ones of the media servers; and

determining whether the selected source location is a disk, and if the selected source location is a disk, defining the corresponding list of possible destination locations to include predetermined mapped ones of the media servers, and predetermined mapped ones of the archives.

13. A process as recited in claim 7 wherein the scheduling information includes a start date and a start time for copying the selected portion of media data from the selected source location in the network to the selected destination location in the network.

14. A process as recited in claim 1 wherein the operations include encoding operations, and wherein the network further includes at least one multimedia device operative to generate media data, each of the multimedia devices being communicatively coupled with a corresponding encoder one of the media servers which is further operative to selectively activate the corresponding multimedia device, and to encode a selected portion of media data generated by the multimedia device.

15. A process as recited in claim 14 wherein the step of displaying a graphical user interface includes displaying an encoder select interface component providing for the user to select one of the encoder media servers, wherein the input information includes information indicative of a selected multimedia device, and wherein the commands and associated parameters include encoding commands and associated encoding parameters for instructing the selected media server to encode a selected portion of media data received from the selected media device.

16. A process as recited in claim 15 wherein the steps of displaying a graphical user interface, receiving information, and generating commands further comprise the steps of:

displaying encoding operation scheduling interface components enabling the user to select a start time and a start date for a delayed encoding operation;

receiving information indicative of a selected start time and a selected start date for initiating an encoding operation; and

generating encoding commands and associated parameters for instructing the selected media server to encode media data received from the selected multimedia device at the selected start time on the selected start date.

17. A process as recited in claim 16 wherein the steps of displaying the encoding operation scheduling components, and receiving information further include the steps of:

displaying duration interface components enabling the user to select from time duration specification options including,

selecting a scheduled stop date and stop time for terminating the encoding operation, and

selecting a time duration for which the encoding operation is to continue following the selected start time on the selected start date; and

receiving information indicative of a selected time duration specification option.

18. A process as recited in claim 15 wherein the operations further include recording operations for recording selected portions of encoded media data that are encoded during corresponding ones of the encoding operations, and wherein the steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying a record-to selection interface component enabling the user to select a storage location by browsing a list of available storage locations including predetermined mapped ones of a plurality of memory devices associated with the selected encoder media server; and

receiving information indicative of the selected storage location;

wherein the commands and associated parameters further include a record command and associated record parameters for instructing the selected encoder media server to store the selected portion of encoded media data at the selected storage location.

19. A process as recited in claim 18 wherein the steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying an asset name selection interface component enabling the user specify a name for the encoded portion of media data to be stored at the selected storage location;

receiving information indicative of a specified asset name; and

wherein the record commands and associated record parameters provide for instructing the selected encoder media server to store the encoded portion of media data at the selected storage location under the specified asset name.

20. A process as recited in claim 19 wherein the steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying an overwriting option selection interface component enabling the user to select from a plurality of overwriting options for specifying attributes of the recording operation if a file having the specified asset name exists at the selected storage location at the time of execution of the recording operation, the overwriting options including,

a prompt option for specifying that the user is to be prompted if a file having the specified asset name exists at the selected storage location at the time of execution of the recording operation,

an overwrite option for specifying that any file having the specified asset name, existing at the selected storage location at the time of execution of the recording operation, is to be overwritten with the encoded portion of media data, and

an overwrite option for specifying that any file having the specified asset name, existing at the selected storage location at the time of execution of the recording operation, is to be appended with the encoded portion of media data.

21. A process as recited in claim 15 wherein at least one of the media servers is further operative to stream media data to selected ones of a plurality of end user terminals via the network, and wherein the operations further include real time streaming operations for streaming the encoded portion of media data from the selected encoder media server to corresponding selected ones of the end user terminals via the network.

22. A process as recited in claim 21 wherein the steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying streaming destination selection interface components enabling the user to select at least one of the end user terminals as a destination for streaming the encoded portion of media data; and

receiving information indicative of at least one selected end user;

wherein the commands and associated parameters further include a streaming command and associated streaming parameters for instructing the selected encoder media server to stream the encoded portion of media data to the selected end users.

23. A process as recited in claim 22 wherein the streaming destination selection interface components include:

manual destination selection interface components enabling the user to select a destination internet protocol (IP) address and port number associated with a selected one of the end user terminals; and

predefined destination selection interface components enabling the user to select a group from a plurality of predefined groups of destination end user terminals, each of the user terminals having an associated IP address and port number.

24. A process as recited in claim 18 wherein at least one of the media servers is further operative to stream media data to selected ones of a plurality of end user terminals via the network, and wherein the operations further include playback operations for streaming the stored portion of encoded media data from the selected encoder media server to corresponding selected ones of a plurality of end user terminals via the network.

25. A process as recited in claim 24 wherein the steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying playback destination selection interface components enabling the user to select at least one of the end user terminals as a destination for streaming the encoded portion of media data in accordance with a user defined playback schedule;

displaying play-back schedule interface components enabling the user define a play-back schedule; and

receiving information indicative a user defined play-back schedule;

wherein the commands and associated parameters further include a play-back command and associated play-back parameters for instructing the selected encoder media server to stream the stored portion of encoded media data to the selected end users via the network in accordance with the user defined play-back schedule.

26. A process as recited in claim 25 wherein the play-back schedule interface components comprise:

a first group of components enabling the user to select a start time and a start date for the play-back schedule; and

a second group of components enabling the user to select from a plurality of options for specifying a play-back schedule duration.

27. A process as recited in claim 26 wherein the options for specifying a playback schedule duration comprise:

specifying a loop count value for repeating the streaming of the stored portion of media data a number of times equal to the loop count value;

specifying a repeat schedule wherein the streaming of the stored portion of media data is repeated until the stored portion of media data is removed from a schedule list; and

specifying an interval schedule wherein the streaming of the stored portion of media data performed in accordance with a user defined schedule.

28. A process as recited in claim 24 wherein the operations further include notification operations associated with corresponding ones of the playback operations, the notification operations for sending notification messages to selected network addresses associated with selected ones of the end user terminals and the administrator terminal.

29. A process as recited in claim 28 wherein the notification operations include determining success or failure of the streaming specified by the corresponding playback operation, and wherein the steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying notification circumstance specification interface components enabling the user to select from the notification circumstance specification options of,

displaying notification circumstance specification interface components enabling the user to select from notification options including,

a first notification option for sending a notification message to selected network addresses if the corresponding playback operation fails,

a second notification option for sending a notification message to selected network addresses if the corresponding playback operation is successful, and

a third notification option for sending a notification message to selected network addresses whether the corresponding playback operation fails or is successful; and

receiving information indicative a selected notification circumstance option;

wherein the commands and associated parameters further include a notification command and associated notification parameters for instructing the selected media server to send notification messages to the selected network addresses in accordance with the selected notification option .

30. A process as recited in claim 1 wherein at least one of the media servers is operative to stream media data to selected ones of a plurality of end user terminals via the network, and wherein the operations further include multicasting operations for streaming selected portions of media data from selected media servers to corresponding selected ones of the end user terminals via the network.

31. A process as recited in claim 30 wherein the steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying multicasting destination selection interface components enabling the user to select at least one of the end user terminals as a destination for multicasting the selected portion of media data in accordance with a user defined multicasting schedule;

displaying multicasting schedule interface components enabling the user define a multicasting schedule; and

receiving information indicative a user defined multicasting schedule;

wherein the commands and associated parameters further include a multicasting command and associated multicasting parameters for instructing the selected media server to stream the

selected portion of media data to the selected end users via the network in accordance with the user defined multicasting schedule.

32. A process as recited in claim 31 wherein the multicasting schedule interface components comprise:

a first group of components enabling the user to select a start time and a start date for a multicasting schedule; and

a second group of components enabling the user to select from a plurality of options for specifying a multicasting schedule duration.

33. A process as recited in claim 32 wherein the options for specifying a multicasting schedule comprise:

specifying a loop count value for repeating the streaming of the stored portion of media data a number of times equal to the loop count value;

specifying a repeat schedule wherein the streaming of the stored portion of media data is repeated until the stored portion of media data is removed from a schedule list; and

specifying an interval schedule wherein the streaming of the stored portion of media data performed in accordance with a user defined schedule.

34. A process as recited in claim 30 wherein the operations further include notification operations associated with corresponding ones of the multicasting operations, the notification operations for sending notification messages to selected network addresses associated with selected ones of the end user terminals and the administrator terminal.

35. A process as recited in claim 34 wherein the notification operations include determining success or failure of the streaming specified by the corresponding multicasting operation, and wherein the steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying notification circumstance specification interface components enabling the user to select from the notification options including,

a first notification option for sending a notification message to selected network addresses if the corresponding multicasting operation fails,

a second notification option for sending a notification message to selected network addresses if the corresponding multicasting operation is successful, and

a third notification option for sending a notification message to selected network addresses if the corresponding multicasting operation fails or is successful; and

receiving information indicative a selected notification circumstance option;

wherein the commands and associated parameters further include a notification command and associated notification parameters for instructing the selected media server to send notification messages to the selected network addresses in accordance with the selected notification option .

36. A process as recited in claim 1 wherein the input information includes server identification information specifying the selected media server, operation information specifying the operation to be performed by the selected media server, and schedule information specifying the schedule, and wherein the process further comprises the step of displaying graphical indicia indicative of a plurality of user scheduled operations that have been previously defined and scheduled by the user, each of the scheduled operations being associated with corresponding previously specified information including previously specified server identification information, previously specified input information, and previously specified schedule information.

37. A process as recited in claim 36 further comprising the steps of:

providing interface components allowing the user to select one of the displayed scheduled operations;

receiving information indicative of a selected scheduled operation;

determining a portion of the interface components of the graphical user interface that is associated with the selected scheduled operation; and

displaying the determined portion of the interface components having previously specified information associated with the selected scheduled operation displayed therein, the displayed portion of the interface components enabling the user to edit the previously specified information associated with the selected scheduled operation in order to generate revised information associated with a revised operation, the revised information including revised server identification information, revised operation information, and revised schedule information.

38. A process as recited in claim 37 further comprising the steps of:

linking to a previously specified media server indicated by previously specified server identification information associated with the selected scheduled operation;

generating delete commands and associated parameters for instructing the previously specified media server to cancel execution of the selected scheduled operation; and

transmitting the delete commands and associated parameters to the previously specified media server.

39. A process as recited in claim 38 further comprising the steps of:

generating new commands and associated parameters based on the revised information associated with the revised operation, the new commands and associated parameters for instructing a particular media server to execute the revised operation in accordance with the revised schedule information; and

transmitting the commands and the associated parameters to the particular media server via the network.

40. A process as recited in claim 36 wherein the network further includes a global asset management server communicatively coupled to the administrator terminal via the network, the

global asset management server also being communicatively coupled to a memory device for storing global information including server identification information, operation information, and schedule information associated with corresponding scheduled operations that have been defined and scheduled by corresponding users, the process further comprising the steps of:

transmitting information indicative of a particular user from the administrator terminal to the management server; and

transmitting the server identification information, the operation information, and the schedule information associated with the specified operation to the management server for storage.

41. A process as recited in claim 40 wherein the management server is further operative to access the global information, the process further comprising the steps of:

accessing the management server from the administrator terminal;

transmitting user identification information indicative of a specified user to the asset management server; and

receiving previously specified information associated with each user scheduled operation that has been previously defined and scheduled by the specified user.

42. A process as recited in claim 40 wherein the global asset management server is further operative to generate a global schedule document including the graphical indicia indicative of the user scheduled operations, the previously specified information corresponding with each of the graphical indicia, executable code for implementing the steps of displaying a graphical user interface, receiving information, generating commands and associated, and transmitting the commands and the associated parameters, and active areas for initiating the executable code, and wherein the step of displaying graphical indicia indicative of a plurality of user scheduled operations further comprises:

transmitting information indicative of a particular user to the asset management server; and

downloading the global schedule document associated with the particular user from the asset management server to the administrator terminal.

43. A process as recited in claim 42 wherein the document comprises a HTML document and wherein the executable code comprises a Java applet.

44. A machine readable storage device having stored therein encoding instructions for executing a process of providing a centralized user interface at an administrator terminal in a network for scheduling and coordinating the execution of operations performed on media data by selected ones of a plurality of media servers of the network, the process comprising the steps of:

displaying a graphical user interface at the administrator terminal, the interface including a plurality of interface components enabling a user to specify and schedule operations to be performed by selected ones of the media servers;

receiving information input by the user identifying a selected one of the media servers, an operation to be performed by the selected media server, and a schedule for performing the operation;

generating commands and associated parameters based on the input information for instructing the selected media server to execute the specified operation in accordance with the specified schedule; and

transmitting the commands and associated parameters to the selected media server.

45. A machine readable storage device as recited in claim 44 wherein the administrator terminal also includes a processing unit, and a browser application executed by the processing unit, and wherein the steps of displaying, receiving, generating, and transmitting are performed by executing an applet over the processing unit.

46. A machine readable storage device as recited in claim 44 wherein at least one of the media servers is further operative to encode media data received from a corresponding multimedia device, and wherein the operations include:

delete operations for deleting portions of media data from a source location selected from locations including predetermined mapped ones of the media servers and the memory devices;

copy operations for copying portions of media data from selected source ones of the locations to selected destination ones of the locations;

multicasting operations for streaming portions of media data from selected ones of the media servers to selected ones of a plurality of end user terminals via the network; and

encoding operations for encoding media data received by selected ones of the media servers.

47. A machine readable storage device as recited in claim 44 wherein each of the media servers is operative to access at least one corresponding memory device for storing media data, and is also operative to stream media data to selected locations via the network, and wherein the operations include copy operations for copying selected portions of media data from selected source locations in the network to selected destination locations in the network.

48. A machine readable storage device as recited in claim 47 wherein the steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying a source selection interface component enabling the user to select a source location by browsing a list of available locations including predetermined mapped ones of the media servers and predetermined mapped ones of the memory devices;

receiving information indicative of a selected source location;

displaying a media data selection interface component enabling the user to select a portion of media data by browsing a list of available a portions of media data stored at the selected source location;

receiving information indicative of a selected portion of media data;

determining a corresponding list of possible destination locations associated with the selected source location; and

displaying a destination selection interface component enabling the user to select a destination location by browsing the corresponding list of possible destination locations.

49. A machine readable storage device as recited in claim 44 wherein the operations include encoding operations, and wherein the network further includes at least one multimedia device operative to generate media data, each of the multimedia devices being communicatively coupled with a corresponding encoder one of the media servers which is further operative to selectively activate the corresponding multimedia device, and to encode a selected portion of media data generated by the multimedia device.

50. A machine readable storage device as recited in claim 49 wherein the step of displaying a graphical user interface includes displaying an encoder select interface component providing for the user to select one of the encoder media servers, wherein the input information includes operation information indicative of a selected multimedia device, and wherein the commands and associated parameters include encoding commands and associated encoding parameters for instructing the selected media server to encode a selected portion of media data received from the selected media device.

51. A machine readable storage device as recited in claim 50 wherein at least one of the media servers is operative to stream media data to selected ones of a plurality of end user terminals via the network, and wherein the operations further include real time streaming

operations for streaming the encoded portion of media data from the selected encoder media server to corresponding selected ones of the end user terminals via the network.

52. A machine readable storage device as recited in claim 51 wherein the steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying streaming destination selection interface components enabling the user to select at least one of the end user terminals as a destination for streaming the encoded portion of media data; and

receiving information indicative of at least one selected end user;

wherein the commands and associated parameters further include a streaming command and associated streaming parameters for instructing the selected encoder media server to stream the encoded portion of media data to the selected end users.

53. A machine readable storage device as recited in claim 50 wherein at least one of the media servers is operative to stream media data to selected ones of a plurality of end user terminals via the network, and wherein the operations further include playback operations for streaming the stored portion of encoded media data from the selected encoder media server to corresponding selected ones of the end user terminals via the network.

54. A machine readable storage device as recited in claim 53 wherein the operations further include notification operations associated with corresponding ones of the playback operations, the notification operations for sending notification messages to selected network addresses associated with selected ones of the end user terminals and the administrator terminal.

55. A machine readable storage device as recited in claim 44 wherein at least one of the media servers is operative to stream media data to selected ones of a plurality of end user terminals via the network, and wherein the operations further include multicasting operations for

streaming selected portions of media data from selected media servers to corresponding selected ones of the end user terminals.

56. A machine readable storage device as recited in claim 55 wherein the steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying multicasting destination selection interface components enabling the user to select at least one of the end user terminals as a destination for multicasting the selected portion of media data in accordance with a user defined multicasting schedule;

displaying multicasting schedule interface components enabling the user define a multicasting schedule; and

receiving information indicative a user defined multicasting schedule;

wherein the commands and associated parameters further include a multicasting command and associated multicasting parameters for instructing the selected media server to stream the selected portion of media data to the selected end users via the network in accordance with the user defined multicasting schedule.

57. A machine readable storage device as recited in claim 56 wherein the operations further include notification operations associated with corresponding ones of the multicasting operations, the notification operations for sending notification messages to selected network addresses associated with selected ones of the end user terminals and the administrator terminal.

58. A machine readable storage device as recited in claim 44 further comprising the step of displaying graphical indicia indicative of a plurality of user scheduled operations that have been previously defined and scheduled by the user, each of the scheduled operations being associated with corresponding previously specified input information including previously specified server identification information, previously specified operation information, and previously specified schedule information.

59. A machine readable storage device as recited in claim 58 further comprising the steps of:

providing interface components allowing the user to select one of the displayed scheduled operations;

receiving information indicative of a selected scheduled operation;

determining a portion of the interface components of the graphical user interface that is associated with the selected scheduled operation; and

displaying the determined portion of the interface components having previously specified information associated with the selected scheduled operation displayed therein, the displayed portion of the interface components enabling the user to edit the previously specified information associated with the selected scheduled operation in order to generate revised information associated with a revised operation, the revised information including revised server identification information, revised operation information, and revised schedule information.

60. A machine readable storage device as recited in claim 59 further comprising the steps of:

linking to a previously specified media server indicated by previously specified server identification information associated with the selected scheduled operation;

generating delete commands and associated parameters for instructing the previously specified media server to cancel execution of the selected scheduled operation; and

transmitting the delete commands and associated parameters to the previously specified media server.

61. A machine readable storage device as recited in claim 60 further comprising the steps of:

generating new commands and associated parameters based on the revised information associated with the revised operation, the new commands and associated parameters for instructing a particular media server to execute the revised operation in accordance with the revised schedule information; and

transmitting the commands and the associated parameters to the particular media server via the network.

62. A server operative to provide an applet to a client via a network, the applet including encoding instructions for executing a process of providing a centralized user interface at the client for scheduling and coordinating the execution of operations performed on media data by selected ones of a plurality of media servers of the network, comprising the steps of:

displaying a graphical user interface at the client, the interface including a plurality of interface components enabling a user to specify and schedule operations to be performed by selected ones of the media servers;

receiving information input by the user identifying a selected one of the media servers, an operation to be performed by the selected media server, and a schedule for performing the operation;

generating commands and associated parameters based on the input information for instructing the selected media server to execute the specified operation in accordance with the specified schedule; and

transmitting the commands and associated parameters to the selected media server.

63. A server as recited in claim 62 wherein at least one of the media servers is further operative to encode media data received from a corresponding multimedia device, and wherein the operations include:

delete operations for deleting portions of media data from catalogs of predetermined mapped ones of the media servers;

copy operations for copying portions of media data from selected source locations in the network to selected destination locations in the network;

multicasting operations for streaming portions of media data from selected ones of the media servers to selected ones of a plurality of end user terminals via the network; and

encoding operations for encoding media data received by selected ones of the media servers.

64. A server as recited in claim 62 wherein at least one of the media servers is also operative to encode and parse portions of media data to create media assets, and wherein each of the media servers is further operative to maintain a catalog including a plurality of pointers to catalogued ones of the media assets stored in a local disk included within the media server.

65. A server as recited in claim 62 wherein each of the media servers is operative to access a memory device for storing media data, and also operative to transfer portions of media data to selected locations via the network, and wherein the operations include copy operations for copying selected portions of media data from selected source locations in the network to selected destination locations in the network.

66. A server as recited in claim 65 wherein the steps of displaying a graphical user interface, and receiving input information further comprise the steps of:

displaying a source selection interface component enabling the user to select a source location by browsing a list of available locations including predetermined mapped ones of the media servers and predetermined mapped ones of the memory devices;

receiving information indicative of a selected source location;

displaying a media data selection interface component enabling the user to select a portion of media data by browsing a list of available a portions of media data stored at the selected source location;

receiving information indicative of a selected portion of media data;

determining a corresponding list of possible destination locations associated with the selected source location; and

displaying a destination selection interface component enabling the user to select a destination location by browsing the corresponding list of possible destination locations.

67. A server as recited in claim 66 wherein the steps of displaying a graphical user interface, receiving information, and generating commands further comprise the steps of:

displaying delayed copying interface components enabling the user to select a start time and a start date for a delayed copying operation;

receiving information indicative of a selected start time and a selected start date; and

generating a copy command and associated parameters for instructing the selected media server to copy the selected portion of media data from the selected source location to the selected destination location at the selected start time on the selected start date.

68. A server as recited in claim 65 further comprising the steps of:

displaying a location mapping interface component enabling the user to define a path to a new location in the network;

receiving information indicative of a selected path for accessing the new location; and

adding the new location associated with the selected path to the list of available locations.

69. A server as recited in claim 62 wherein the input information includes server identification information specifying the selected media server, operation information specifying the operation to be performed by the selected media server, and schedule information specifying the schedule, and wherein the process further comprises the step of displaying graphical indicia indicative of a plurality of user scheduled operations that have been previously defined and scheduled by the user, each of the scheduled operations being associated with corresponding previously specified information including previously specified server identification information, previously specified input information, and previously specified schedule information.

70. A server as recited in claim 69 further comprising the steps of:

providing interface components allowing the user to select one of the displayed scheduled operations;

receiving information indicative of a selected scheduled operation;

determining a portion of the interface components of the graphical user interface that is associated with the selected scheduled operation; and

displaying the determined portion of the interface components having previously specified information associated with the selected scheduled operation displayed therein, the displayed portion of the interface components enabling the user to edit the previously specified information associated with the selected scheduled operation in order to generate revised information associated with a revised operation, the revised information including revised server identification information, revised operation information, and revised schedule information.

71. A server as recited in claim 70 further comprising the steps of:

linking to a previously specified media server indicated by previously specified server identification information associated with the selected scheduled operation;

generating delete commands and associated parameters for instructing the previously specified media server to cancel execution of the selected scheduled operation; and

transmitting the delete commands and associated parameters to the previously specified media server.

72. A server as recited in claim 71 further comprising the steps of:

generating new commands and associated parameters based on the revised information associated with the revised operation, the new commands and associated parameters for instructing a particular media server to execute the revised operation in accordance with the revised schedule information; and

transmitting the commands and the associated parameters to the particular media server via the network.

73. A server as recited in claim 69 wherein the network further includes a global asset management server communicatively coupled to the client via the network, the global asset management server also being communicatively coupled to a memory device for storing global information including server identification information, operation information, and schedule information associated with corresponding scheduled operations that have been defined and scheduled by corresponding users, the process further comprising the steps of:

transmitting information indicative of a particular user from the client to the management server; and

transmitting the server identification information, the operation information, and the schedule information associated with the specified operation to the management server for storage.

74. A server as recited in claim 73 wherein the management server is further operative to access the global information, the process further comprising the steps of:

accessing the management server from the client;

transmitting user identification information indicative of a specified user to the asset management server; and

receiving previously specified information associated with each user scheduled operation that has been previously defined and scheduled by the specified user.

75. A server as recited in claim 73 wherein the global asset management server is further operative to generate a global schedule document including the graphical indicia indicative of the user scheduled operations, the previously specified information corresponding with each of the graphical indicia, executable code for implementing the steps of displaying a graphical user

interface, receiving information, generating commands and associated, and transmitting the commands and the associated parameters, and active areas for initiating the executable code, and wherein the step of displaying graphical indicia indicative of a plurality of user scheduled operations further comprises:

transmitting information indicative of a particular user to the asset management server; and

downloading the global schedule document associated with the particular user from the asset management server to the client.

76. A server as recited in claim 75 wherein the document comprises a HTML document and wherein the executable code comprises a Java applet.

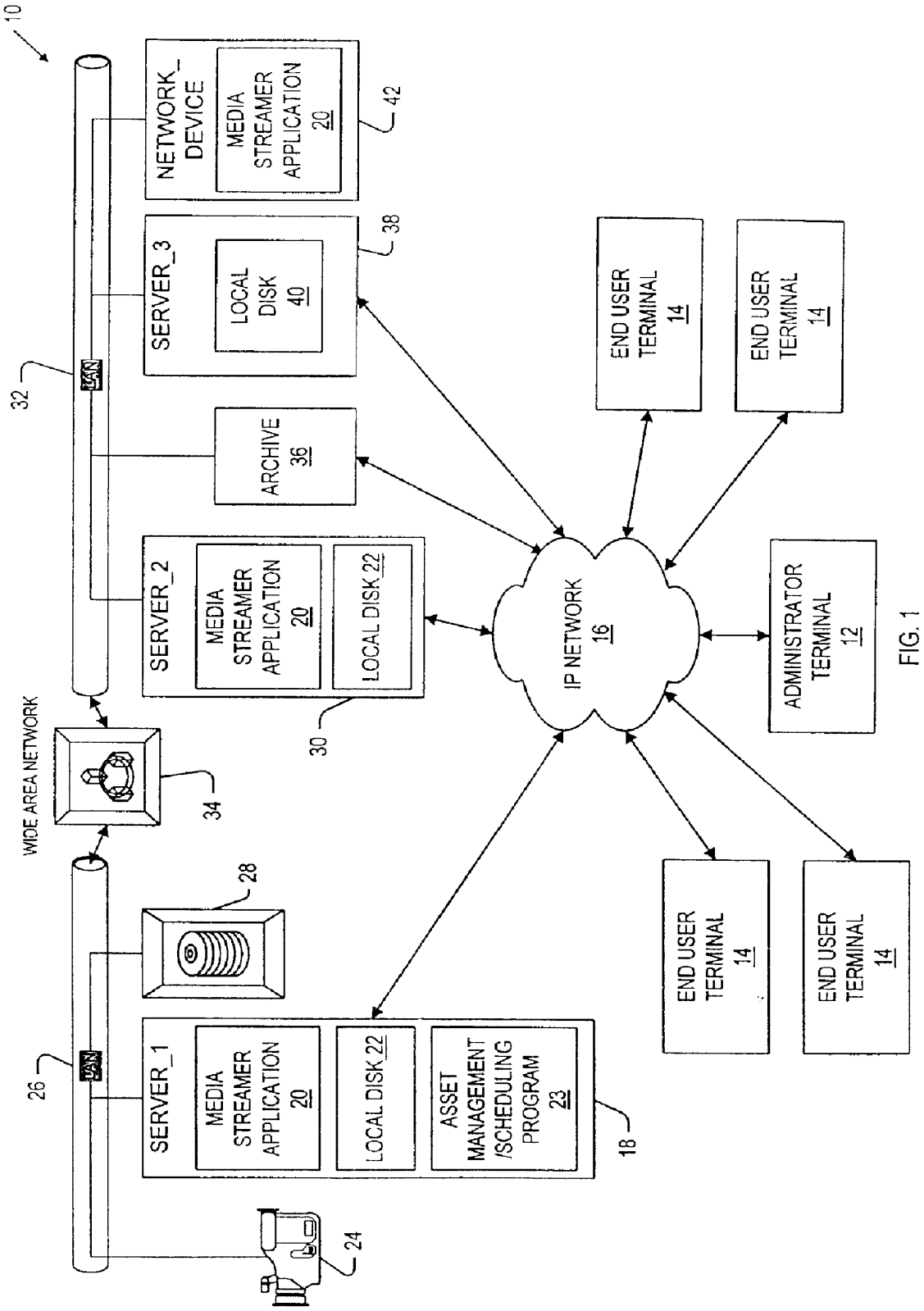

10
NETWORK_ DEVICE
MEDIA STREAMER APPLICATION 20
42
SERVER_3
LOCAL DISK 40
38
32
LAN
ARCHIVE 36
SERVER_2
MEDIA STREAMER APPLICATION 20
LOCAL DISK_22
30
WIDE AREA NETWORK
34
28
26
LAN
SERVER_1
MEDIA STREAMER APPLICATION 20
LOCAL DISK_22
ASSET MANAGEMENT /SCHEDULING PROGRAM 23
18
24
IP NETWORK 16
END USER TERMINAL 14
END USER TERMINAL 14
END USER TERMINAL 14
END USER TERMINAL 14
ADMINISTRATOR TERMINAL 12

FIG. 1

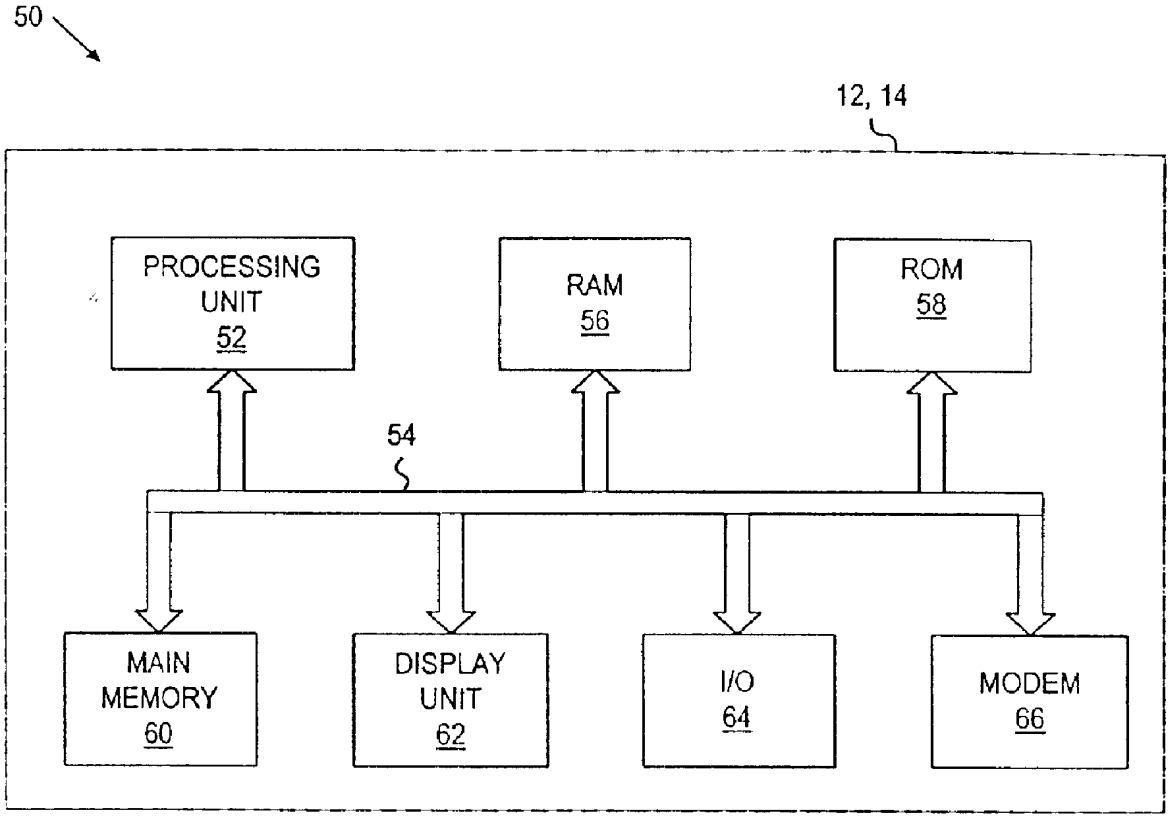

50
12, 14
PROCESSING UNIT 52
RAM 56
ROM 58
54
MAIN MEMORY 60
DISPLAY UNIT 62
I/O 64
MODEM 66

FIG. 2

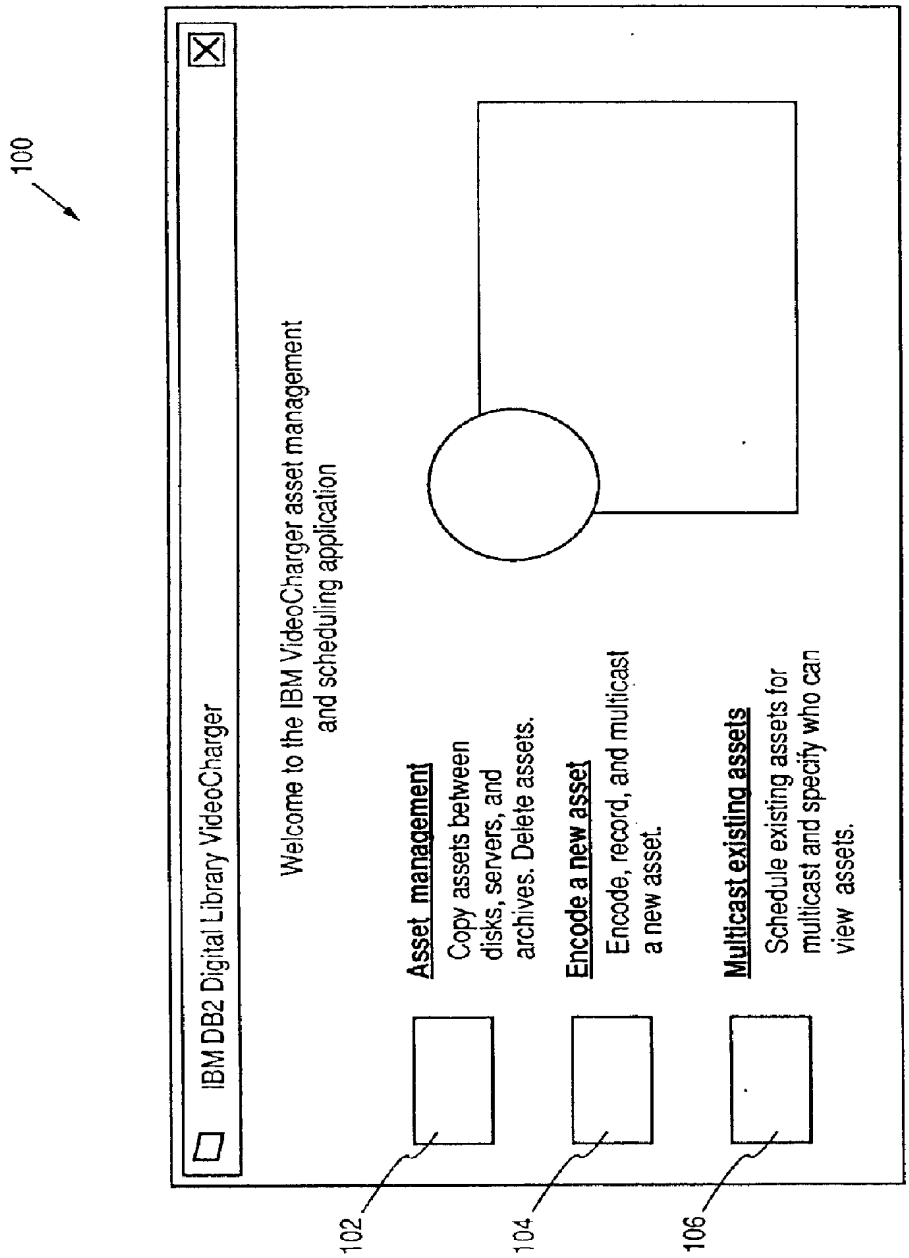

100
IBM DB2 Digital Library VideoCharger
Welcome to the IBM VideoCharger asset management and scheduling application
Asset management
Copy assets between disks, servers, and archives. Delete assets.
Encode a new asset
Encode, record, and multicast a new asset.
Multicast existing assets
Schedule existing assets for multicast and specify who can view assets.
102
104
106

Fig. 3

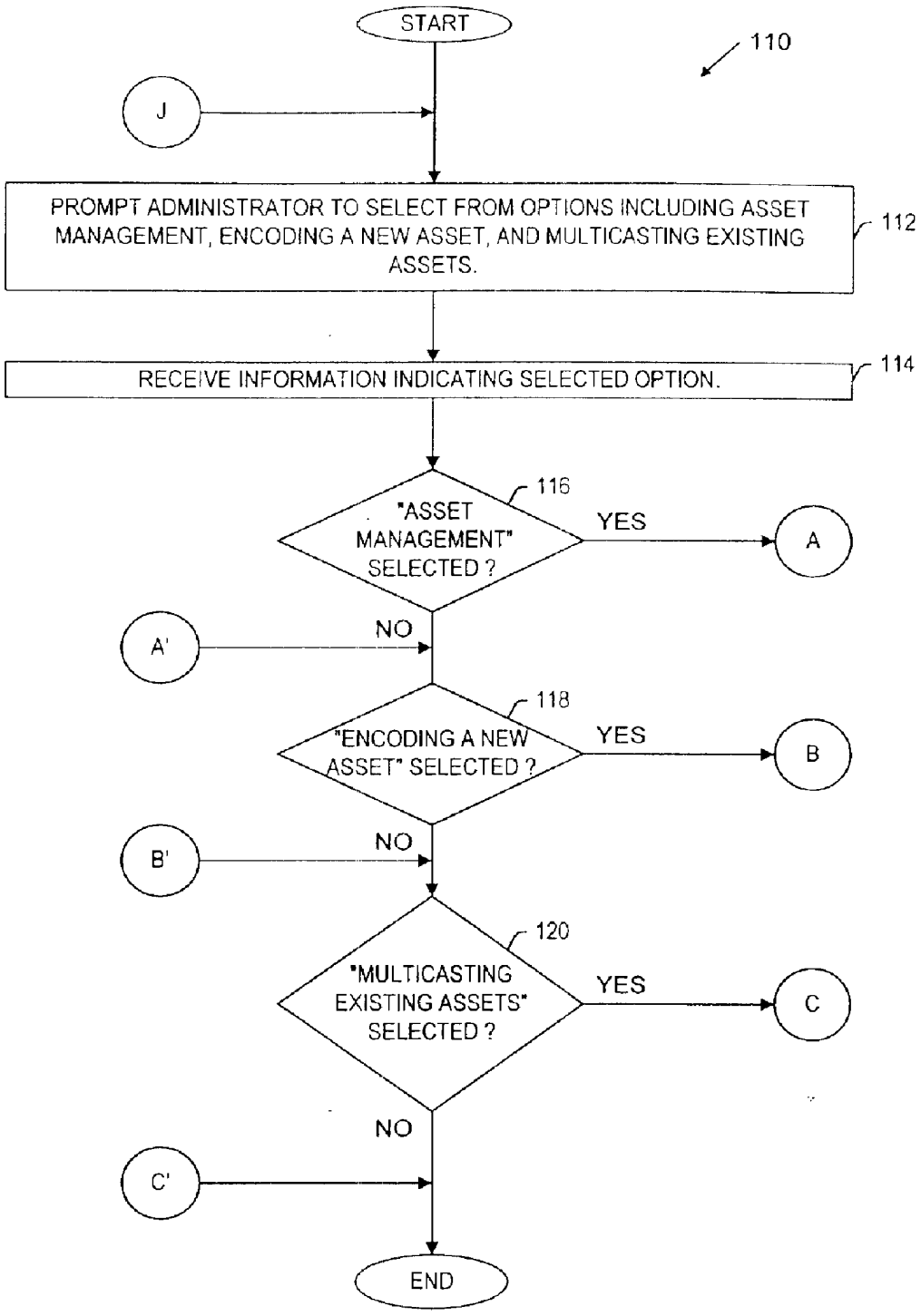

START
110
J
PROMPT ADMINISTRATOR TO SELECT FROM OPTIONS INCLUDING ASSET MANAGEMENT, ENCODING A NEW ASSET, AND MULTICASTING EXISTING ASSETS.
112
RECEIVE INFORMATION INDICATING SELECTED OPTION.
114
116
"ASSET MANAGEMENT" SELECTED ?
YES
A
NO
A'
118
"ENCODING A NEW ASSET" SELECTED ?
YES
B
NO
B'
120
"MULTICASTING EXISTING ASSETS" SELECTED ?
YES
C
NO
C'
END

FIG. 4

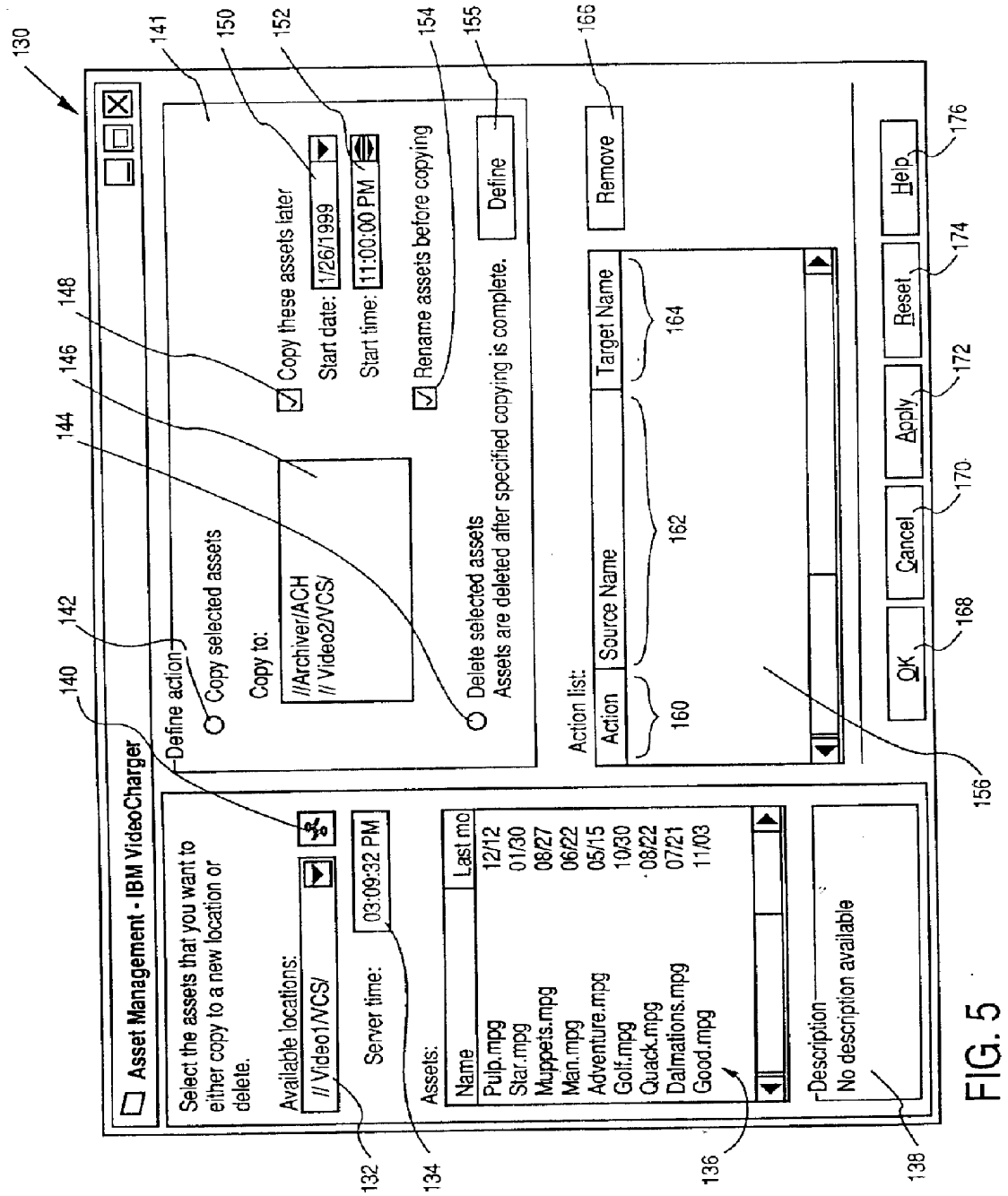

130
Asset Management - IBM VideoCharger
Select the assets that you want to either copy to a new location or delete.
Available locations:
// Video1/VCS/
Server time:
03:09:32 PM
Assets:
Name
Last mo
Pulp.mpg 12/12
Star.mpg 01/30
Muppets.mpg 08/27
Man.mpg 06/22
Adventure.mpg 05/15
Golf.mpg 10/30
Quack.mpg 08/22
Dalmations.mpg 07/21
Good.mpg 11/03
Description
No description available
Define action
Copy selected assets
Copy to:
//Archiver/ACH
// Video2/VCS/
Delete selected assets
Assets are deleted after specified copying is complete.
Copy these assets later
Start date: 1/26/1999
Start time: 11:00:00 PM
Rename assets before copying
Define
Action list:
Action
Source Name
Target Name
Remove
OK
Cancel
Apply
Reset
Help
132
134
136
138
140
141
142
144
146
148
150
152
154
155
156
160
162
164
166
168
170
172
174
176

FIG. 5

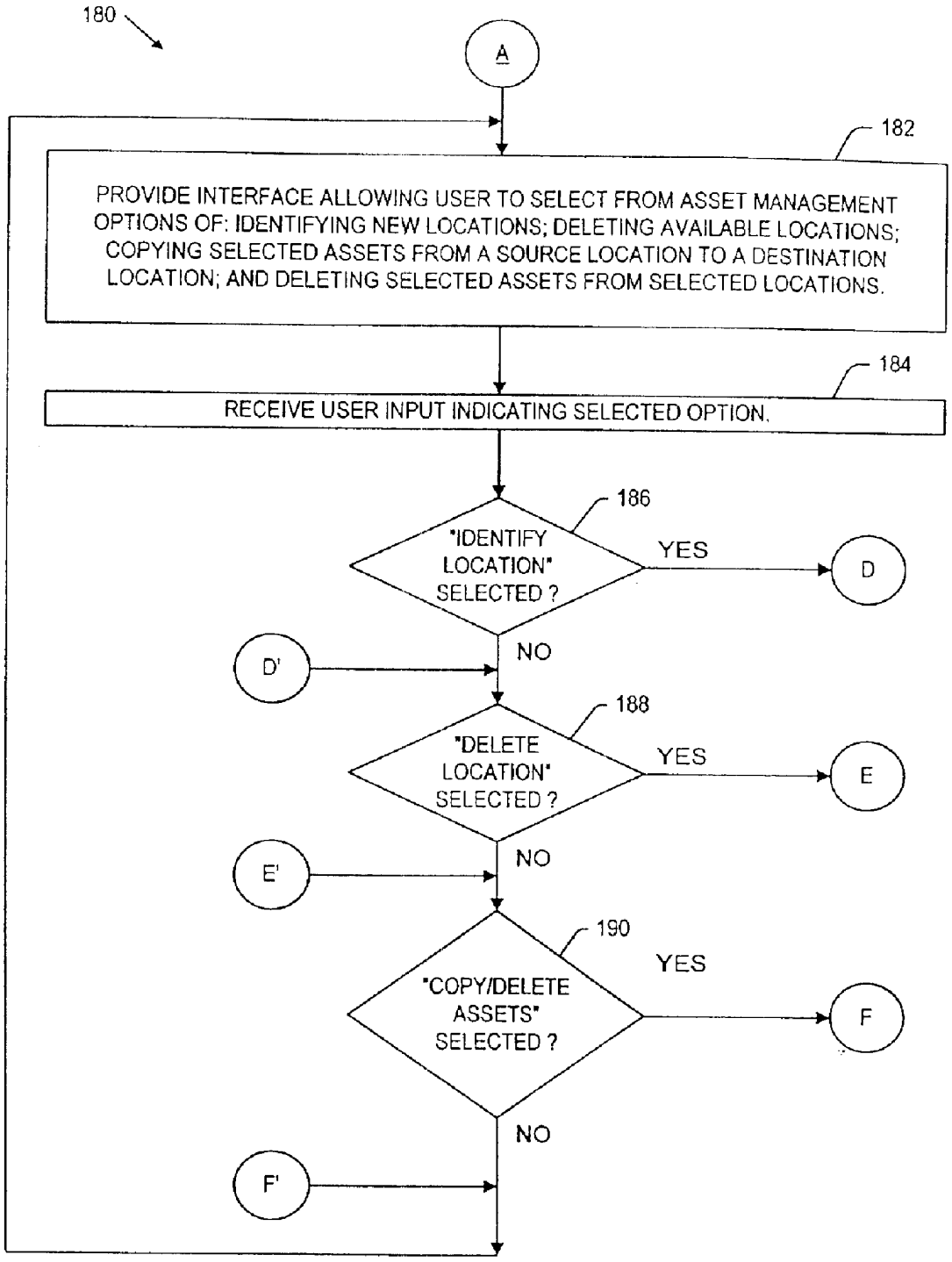

180
A
182
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM ASSET MANAGEMENT OPTIONS OF: IDENTIFYING NEW LOCATIONS; DELETING AVAILABLE LOCATIONS; COPYING SELECTED ASSETS FROM A SOURCE LOCATION TO A DESTINATION LOCATION; AND DELETING SELECTED ASSETS FROM SELECTED LOCATIONS.
184
RECEIVE USER INPUT INDICATING SELECTED OPTION.
186
"IDENTIFY LOCATION" SELECTED ?
YES
D
NO
D'
188
"DELETE LOCATION" SELECTED ?
YES
E
NO
E'
190
"COPY/DELETE ASSETS" SELECTED ?
YES
F
NO
F'

FIG. 6

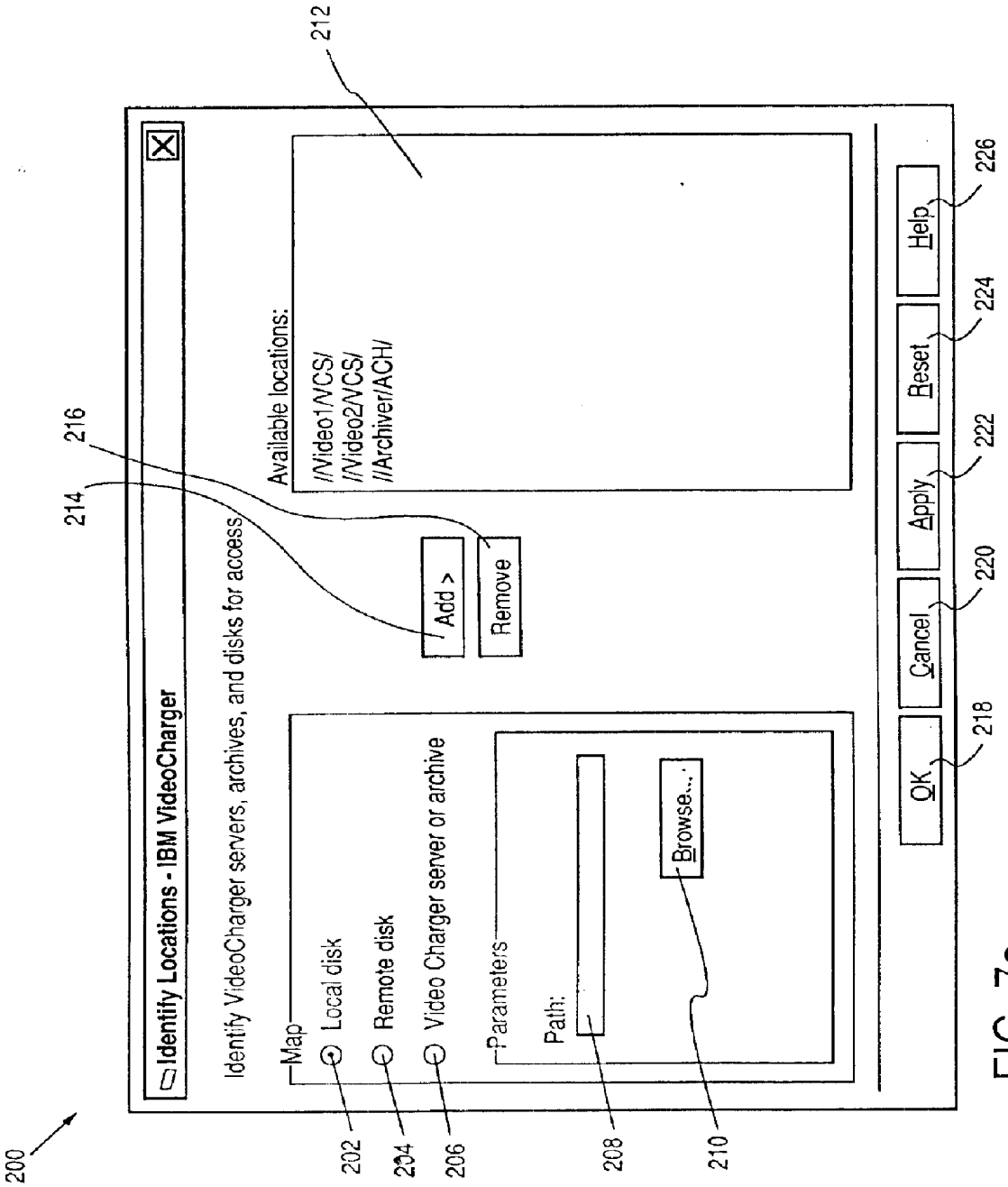

200
Identify Locations - IBM VideoCharger
Identify VideoCharger servers, archives, and disks for access
Map
Local disk
Remote disk
Video Charger server or archive
Parameters
Path:
Browse...
Add >
Remove
Available locations:
//Video1/VCS/
//Video2/VCS/
//Archiver/ACH/
OK
Cancel
Apply
Reset
Help
202
204
206
208
210
212
214
216
218
220
222
224
226

FIG. 7a

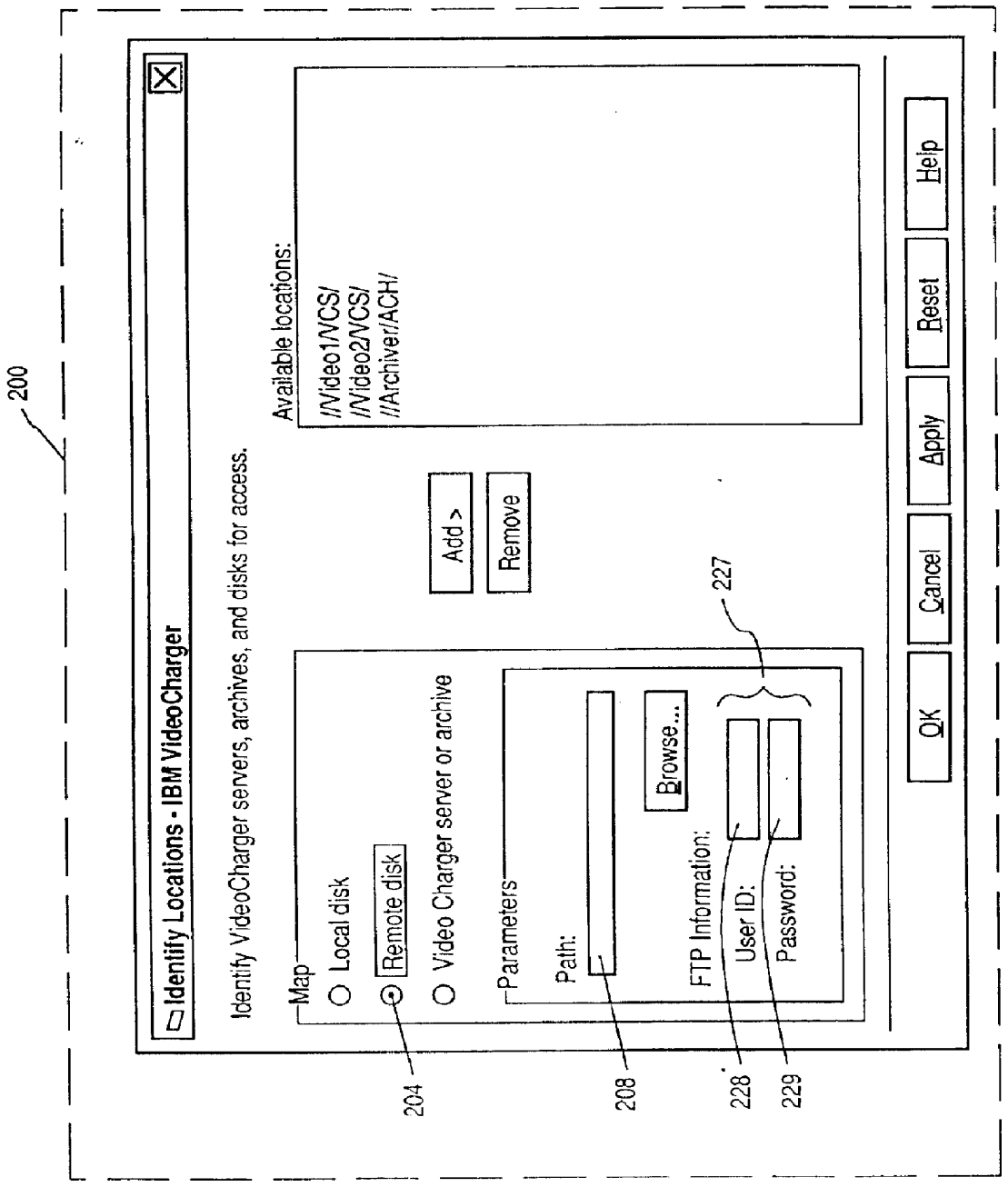

200
Identify Locations - IBM VideoCharger
Identify VideoCharger servers, archives, and disks for access.
Map
Local disk
Remote disk
Video Charger server or archive
Parameters
Path:
Browse...
FTP Information:
User ID:
Password:
Available locations:
//Video1/VCS/
//Video2/VCS/
//Archiver/ACH/
Add >
Remove
OK
Cancel
Apply
Reset
Help
204
208
227
228
229

FIG. 7b

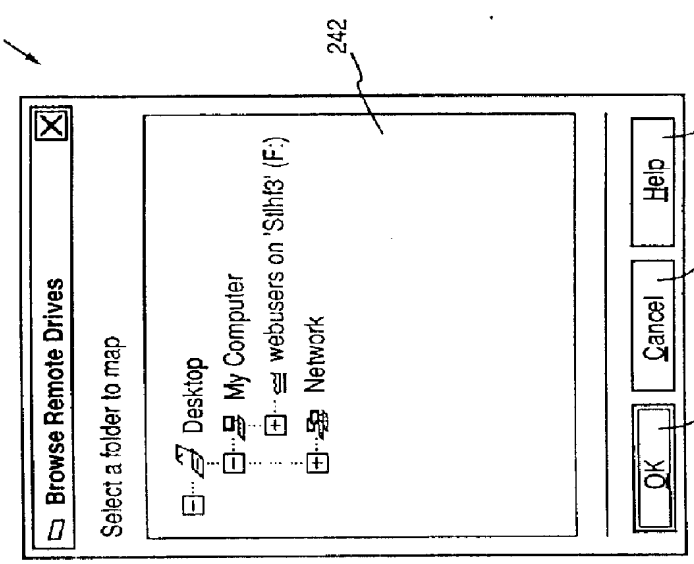

240
Browse Remote Drives
Select a folder to map
242
Desktop
My Computer
webusers on 'Silhf3' (F:)
Network
OK
Cancel
Help
244
246
248

FIG. 7d

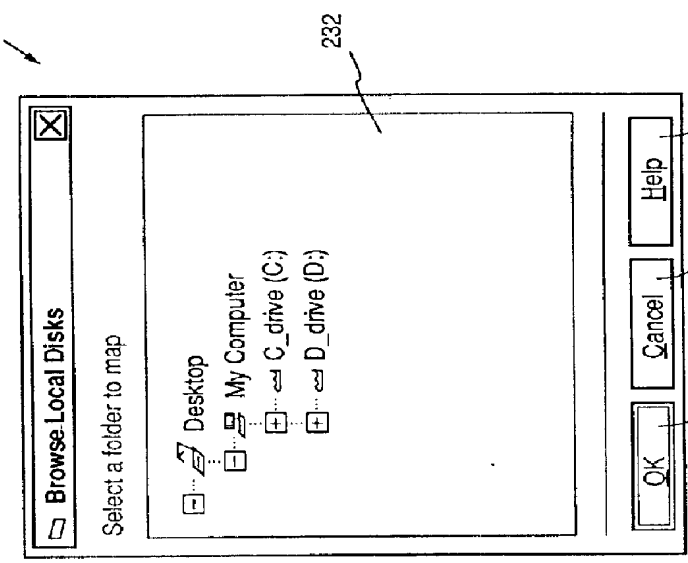

230
Browse Local Disks
Select a folder to map
232
Desktop
My Computer
C_drive (C:)
D_drive (D:)
OK
Cancel
Help
234
236
238

FIG. 7c

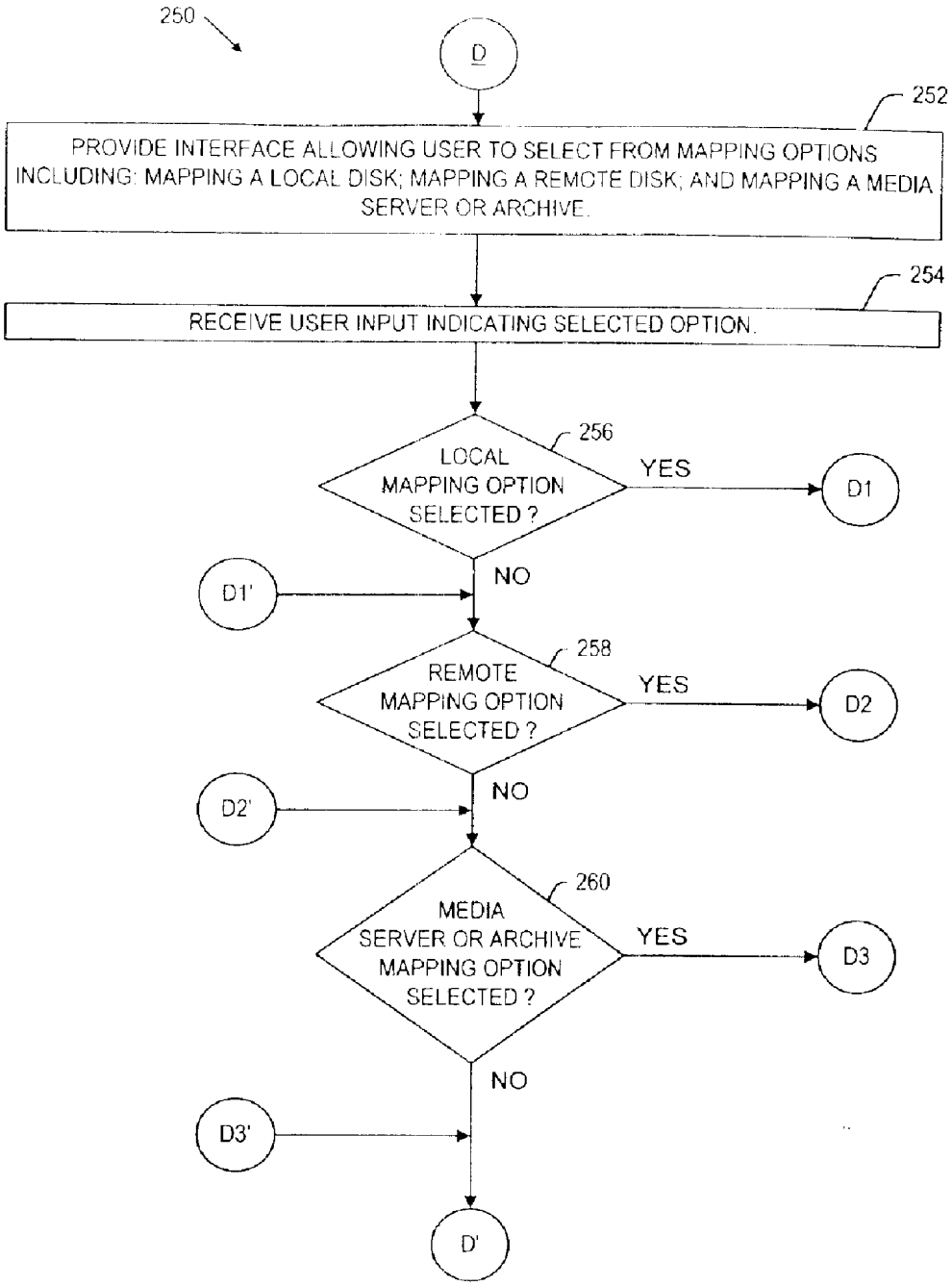

250
D
252
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM MAPPING OPTIONS INCLUDING: MAPPING A LOCAL DISK; MAPPING A REMOTE DISK; AND MAPPING A MEDIA SERVER OR ARCHIVE.
254
RECEIVE USER INPUT INDICATING SELECTED OPTION.
256
LOCAL MAPPING OPTION SELECTED ?
YES
D1
NO
D1'
258
REMOTE MAPPING OPTION SELECTED ?
YES
D2
NO
D2'
260
MEDIA SERVER OR ARCHIVE MAPPING OPTION SELECTED ?
YES
D3
NO
D3'
D'

FIG. 8A

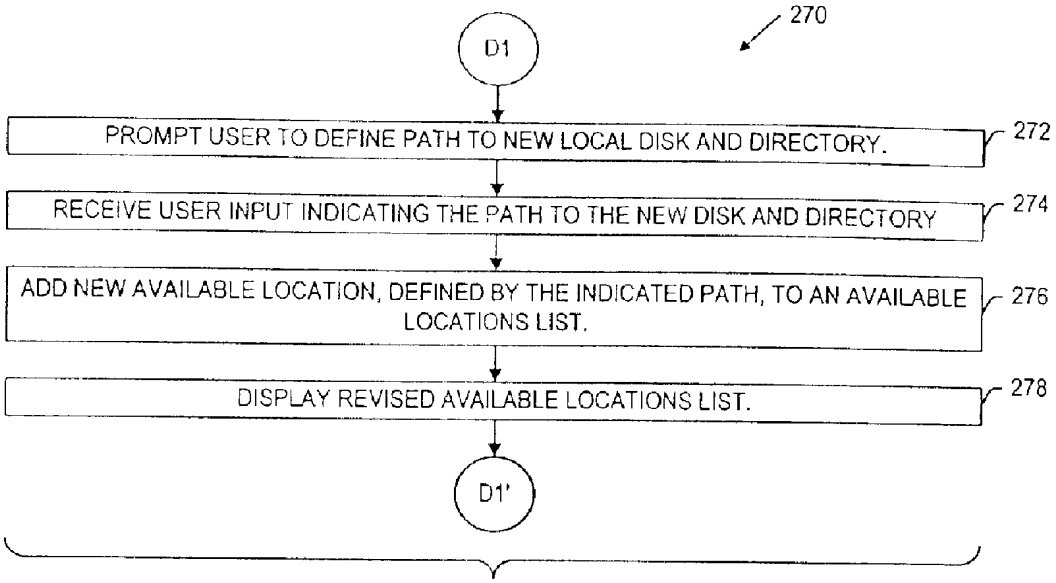

270
D1
PROMPT USER TO DEFINE PATH TO NEW LOCAL DISK AND DIRECTORY.
272
RECEIVE USER INPUT INDICATING THE PATH TO THE NEW DISK AND DIRECTORY
274
ADD NEW AVAILABLE LOCATION, DEFINED BY THE INDICATED PATH, TO AN AVAILABLE LOCATIONS LIST.
276
DISPLAY REVISED AVAILABLE LOCATIONS LIST.
278
D1'

FIG. 8B

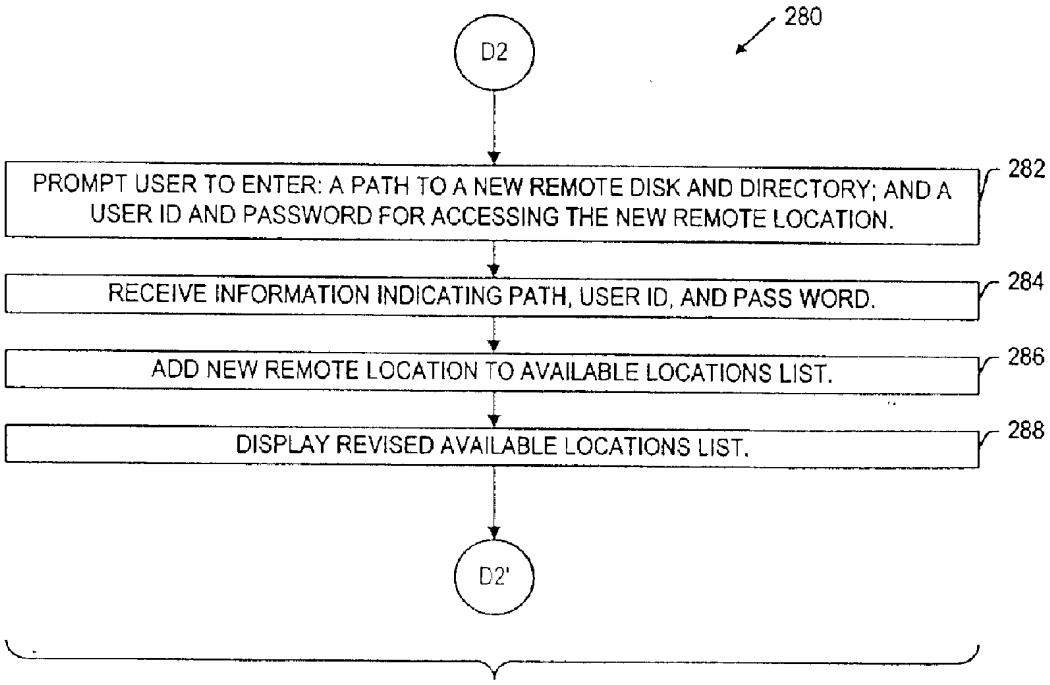

280
D2
PROMPT USER TO ENTER: A PATH TO A NEW REMOTE DISK AND DIRECTORY; AND A USER ID AND PASSWORD FOR ACCESSING THE NEW REMOTE LOCATION.
282
RECEIVE INFORMATION INDICATING PATH, USER ID, AND PASS WORD.
284
ADD NEW REMOTE LOCATION TO AVAILABLE LOCATIONS LIST.
286
DISPLAY REVISED AVAILABLE LOCATIONS LIST.
288
D2'

FIG. 8C

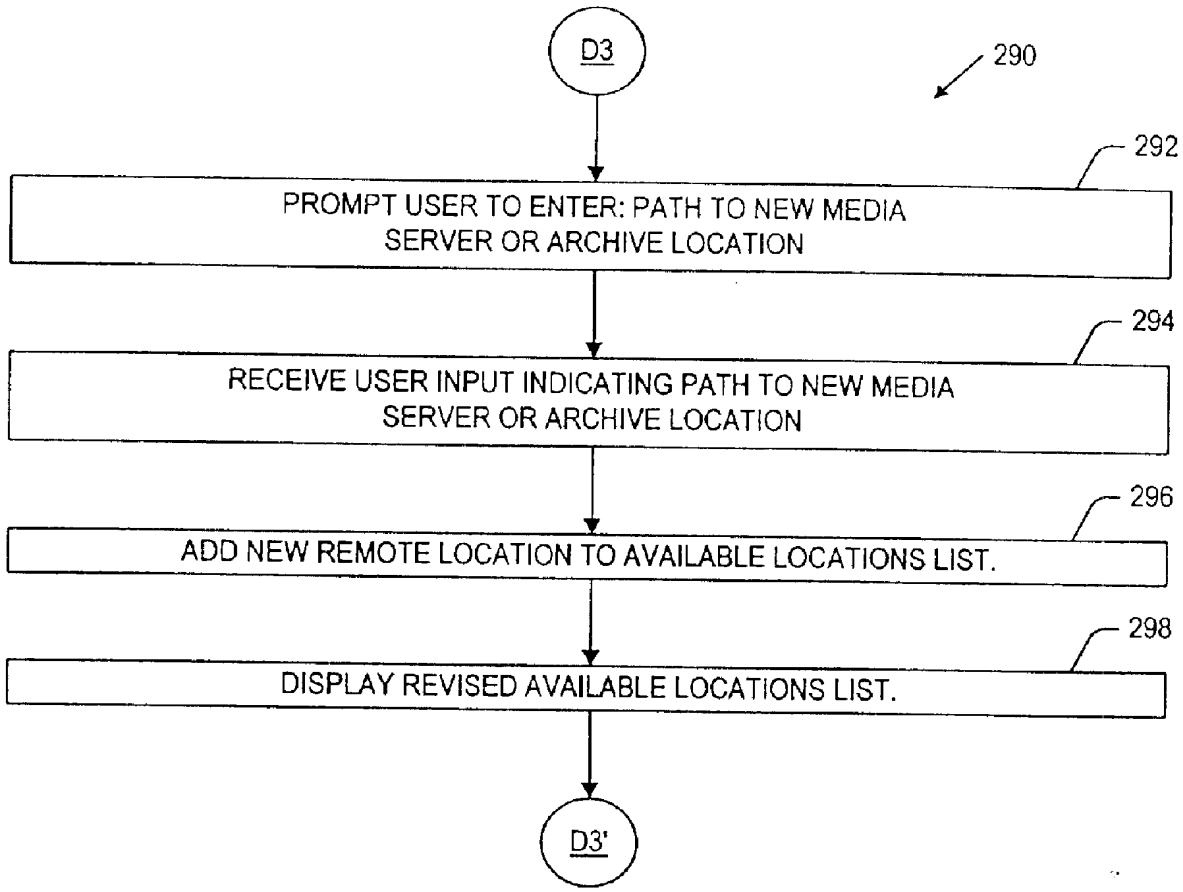

D3
290
292
PROMPT USER TO ENTER: PATH TO NEW MEDIA SERVER OR ARCHIVE LOCATION
294
RECEIVE USER INPUT INDICATING PATH TO NEW MEDIA SERVER OR ARCHIVE LOCATION
296
ADD NEW REMOTE LOCATION TO AVAILABLE LOCATIONS LIST.
298
DISPLAY REVISED AVAILABLE LOCATIONS LIST.
D3'

FIG. 8D

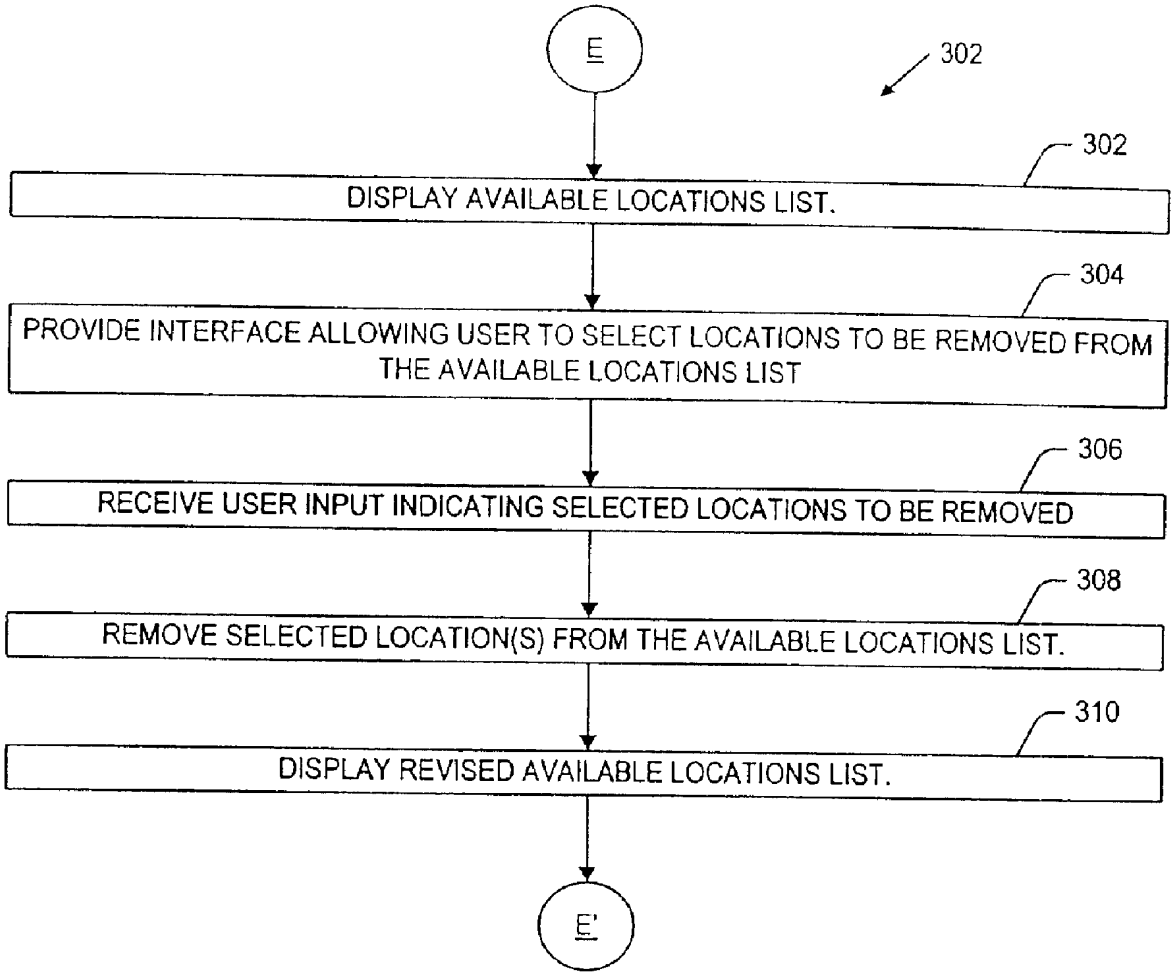

E
302
302
DISPLAY AVAILABLE LOCATIONS LIST.
304
PROVIDE INTERFACE ALLOWING USER TO SELECT LOCATIONS TO BE REMOVED FROM THE AVAILABLE LOCATIONS LIST
306
RECEIVE USER INPUT INDICATING SELECTED LOCATIONS TO BE REMOVED
308
REMOVE SELECTED LOCATION(S) FROM THE AVAILABLE LOCATIONS LIST.
310
DISPLAY REVISED AVAILABLE LOCATIONS LIST.
E'

FIG. 9

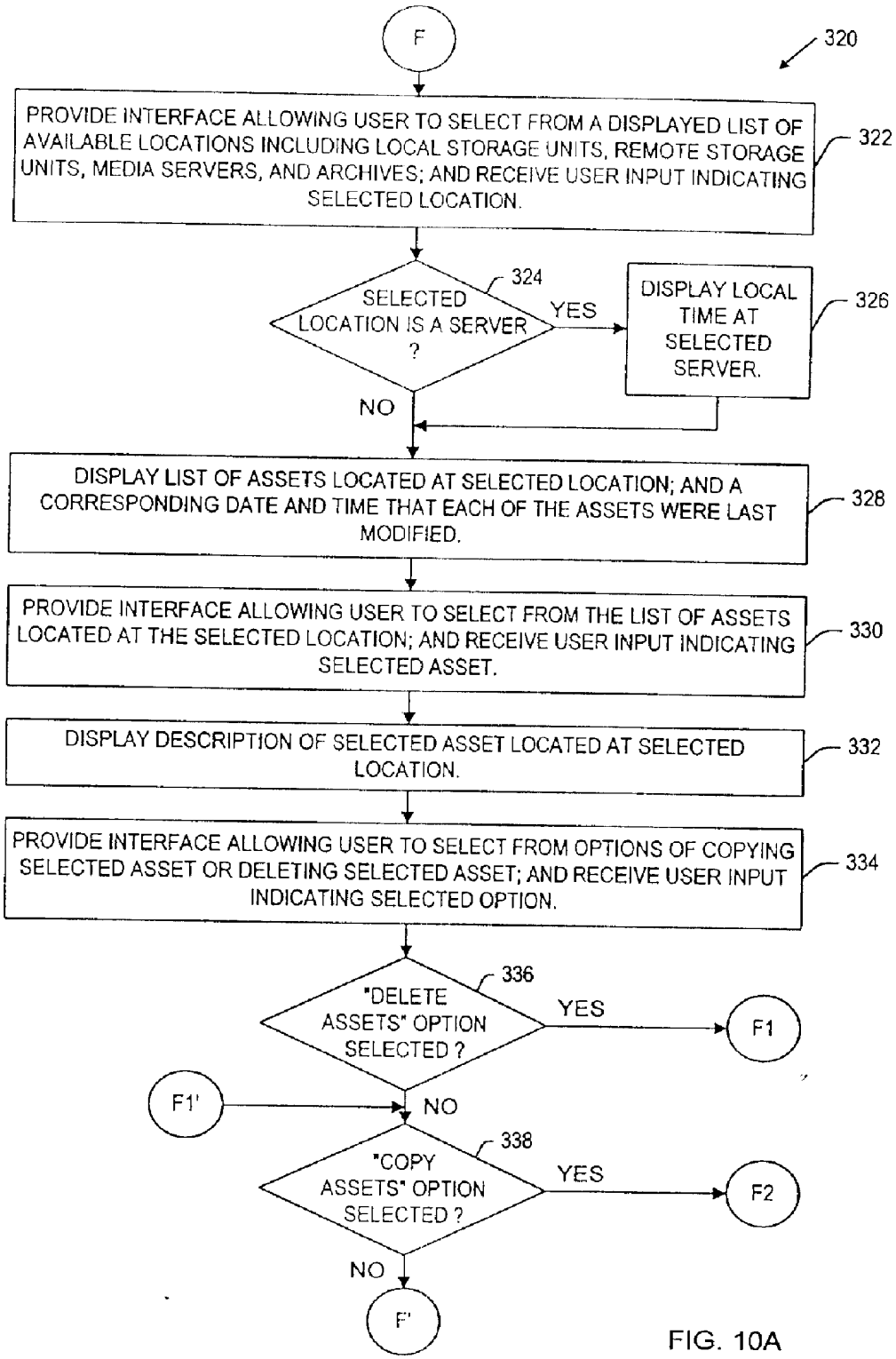

F
320
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A DISPLAYED LIST OF AVAILABLE LOCATIONS INCLUDING LOCAL STORAGE UNITS, REMOTE STORAGE UNITS, MEDIA SERVERS, AND ARCHIVES; AND RECEIVE USER INPUT INDICATING SELECTED LOCATION.
322
SELECTED LOCATION IS A SERVER ?
324
YES
DISPLAY LOCAL TIME AT SELECTED SERVER.
326
NO
DISPLAY LIST OF ASSETS LOCATED AT SELECTED LOCATION; AND A CORRESPONDING DATE AND TIME THAT EACH OF THE ASSETS WERE LAST MODIFIED.
328
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM THE LIST OF ASSETS LOCATED AT THE SELECTED LOCATION; AND RECEIVE USER INPUT INDICATING SELECTED ASSET.
330
DISPLAY DESCRIPTION OF SELECTED ASSET LOCATED AT SELECTED LOCATION.
332
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM OPTIONS OF COPYING SELECTED ASSET OR DELETING SELECTED ASSET; AND RECEIVE USER INPUT INDICATING SELECTED OPTION.
334
"DELETE ASSETS" OPTION SELECTED ?
336
YES
F1
F1'
NO
"COPY ASSETS" OPTION SELECTED ?
338
YES
F2
NO
F'

FIG. 10A

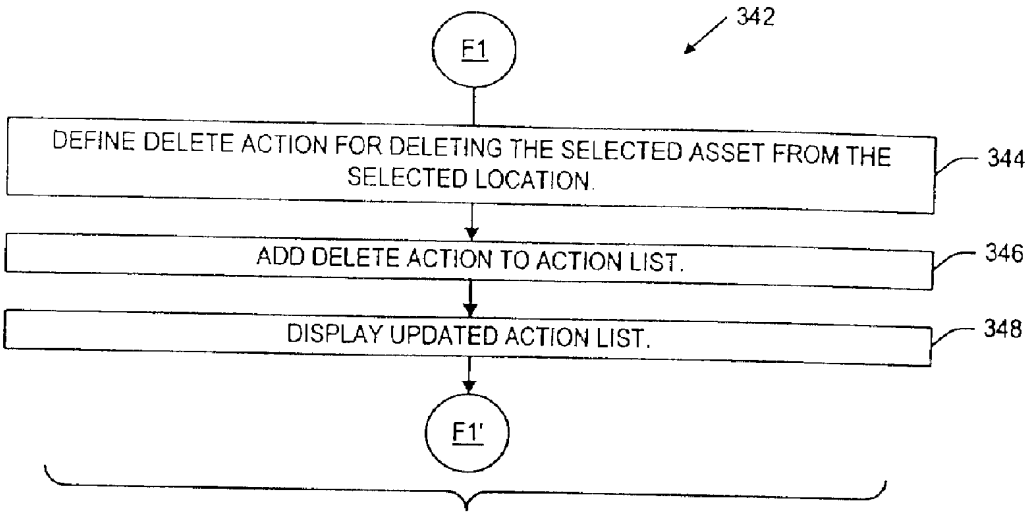


FIG. 10B

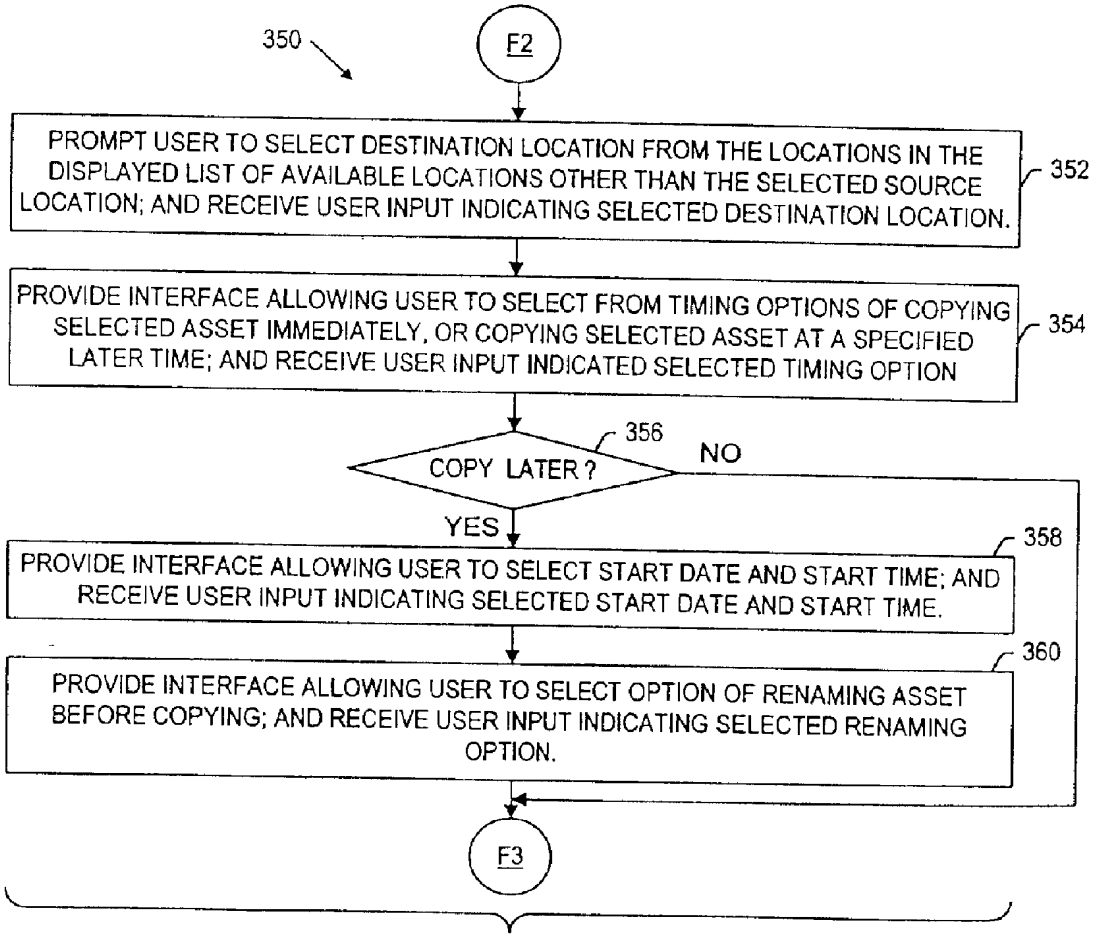


FIG. 10C

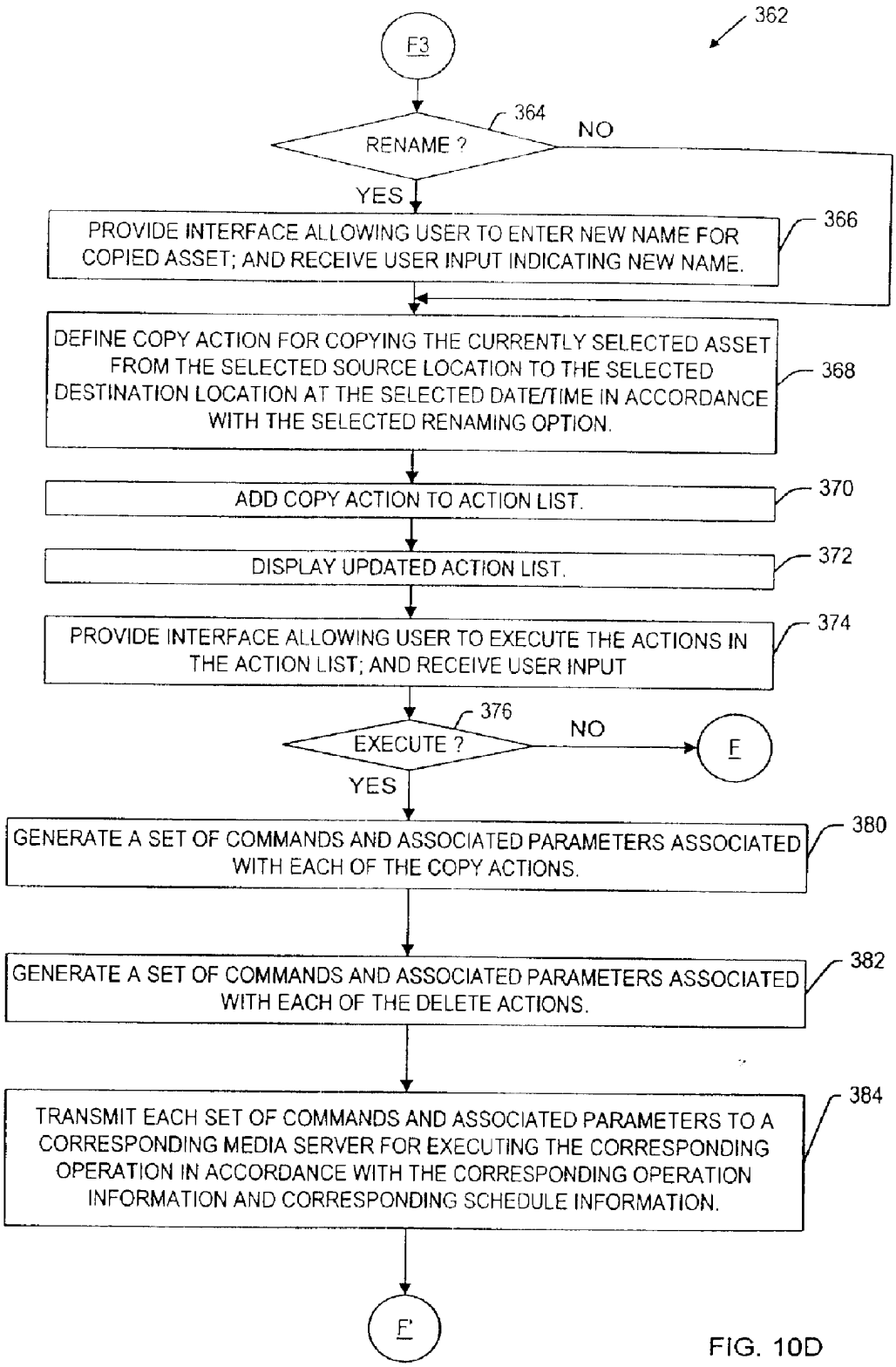

362
F3
364
RENAME ?
NO
YES
PROVIDE INTERFACE ALLOWING USER TO ENTER NEW NAME FOR COPIED ASSET; AND RECEIVE USER INPUT INDICATING NEW NAME.
366
DEFINE COPY ACTION FOR COPYING THE CURRENTLY SELECTED ASSET FROM THE SELECTED SOURCE LOCATION TO THE SELECTED DESTINATION LOCATION AT THE SELECTED DATE/TIME IN ACCORDANCE WITH THE SELECTED RENAMING OPTION.
368
ADD COPY ACTION TO ACTION LIST.
370
DISPLAY UPDATED ACTION LIST.
372
PROVIDE INTERFACE ALLOWING USER TO EXECUTE THE ACTIONS IN THE ACTION LIST; AND RECEIVE USER INPUT
374
376
EXECUTE ?
NO
E
YES
GENERATE A SET OF COMMANDS AND ASSOCIATED PARAMETERS ASSOCIATED WITH EACH OF THE COPY ACTIONS.
380
GENERATE A SET OF COMMANDS AND ASSOCIATED PARAMETERS ASSOCIATED WITH EACH OF THE DELETE ACTIONS.
382
TRANSMIT EACH SET OF COMMANDS AND ASSOCIATED PARAMETERS TO A CORRESPONDING MEDIA SERVER FOR EXECUTING THE CORRESPONDING OPERATION IN ACCORDANCE WITH THE CORRESPONDING OPERATION INFORMATION AND CORRESPONDING SCHEDULE INFORMATION.
384
F'

FIG. 10D

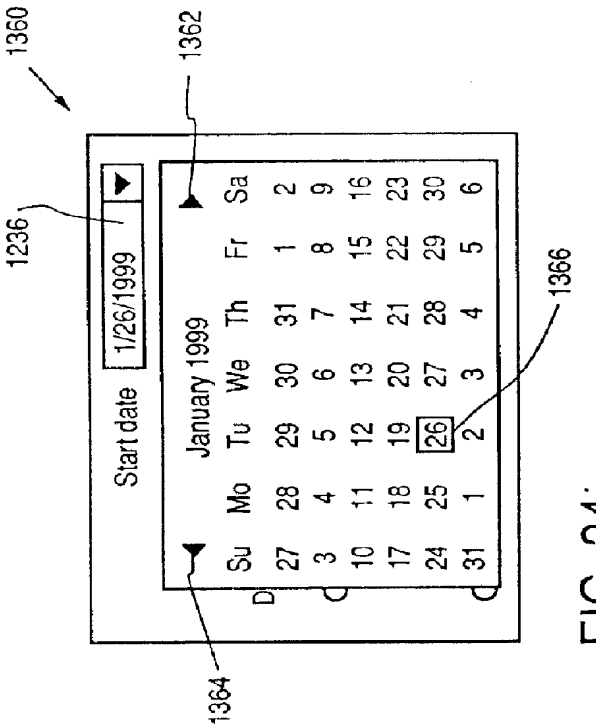

1360
1362
1236
Start date
1/26/1999
January 1999
Su Mo Tu We Th Fr Sa
1364
1366

FIG. 34

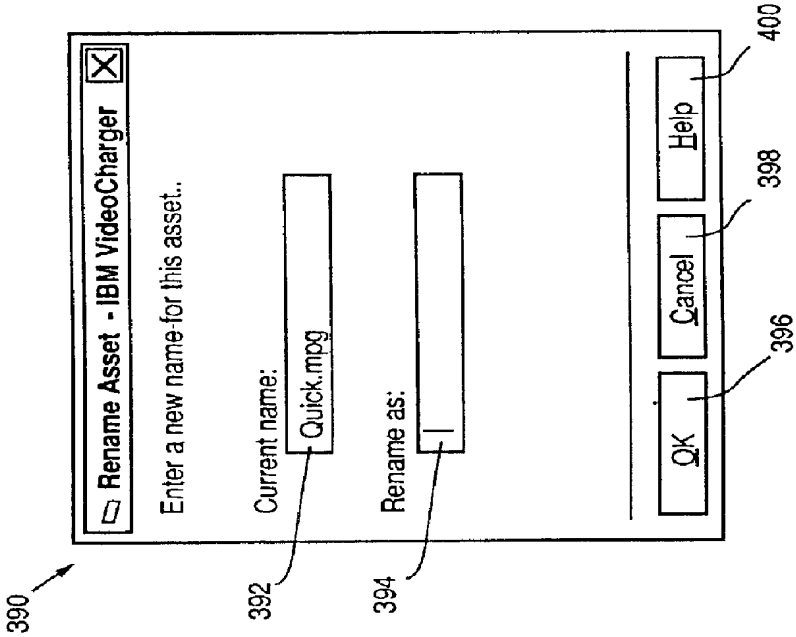

Rename Asset - IBM VideoCharger
Enter a new name for this asset..
Current name:
Quick.mpg
Rename as:
OK
Cancel
Help
390
392
394
396
398
400

FIG. 11

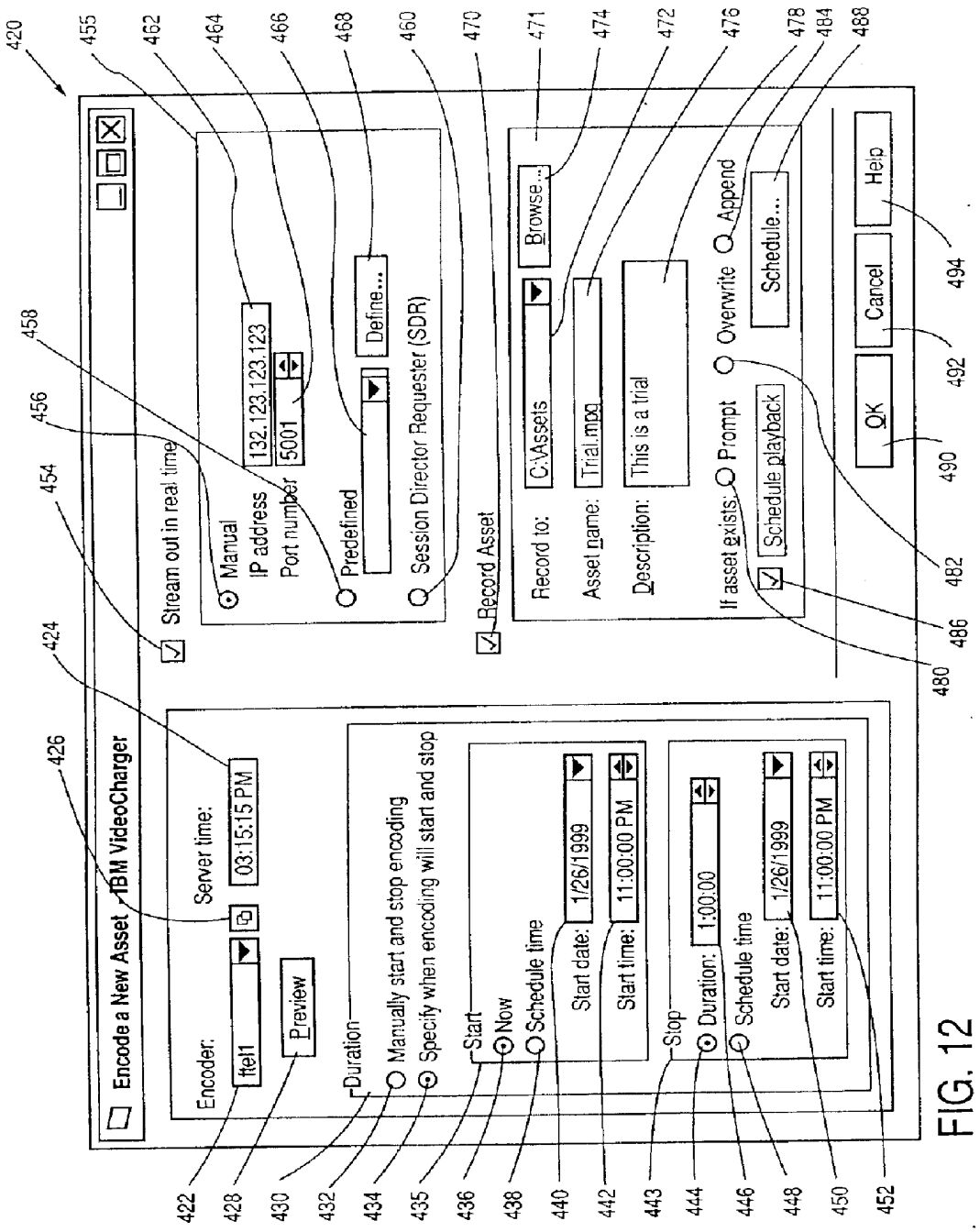

Encode a New Asset - IBM VideoCharger
Encoder:
fte!1
Preview
Server time:
03:15:15 PM
Stream out in real time
Manual
IP address
Port number
132.123.123.123
5001
Predefined
Define...
Session Director Requester (SDR)
Record Asset
Record to:
C:\Assets
Browse...
Asset name:
Trial.mpg
Description:
This is a trial
If asset exists:
Prompt
Overwrite
Append
Schedule playback
Schedule...
OK
Cancel
Help
Duration
Manually start and stop encoding
Specify when encoding will start and stop
Start
Now
Schedule time
Start date:
1/26/1999
Start time:
11:00:00 PM
Stop
Duration:
1:00:00
Schedule time
Start date:
1/26/1999
Start time:
11:00:00 PM
420
422
424
426
428
430
432
434
435
436
438
440
442
443
444
446
448
450
452
454
455
456
458
460
462
464
466
468
470
471
472
474
476
478
480
482
484
486
488
490
492
494

FIG. 12

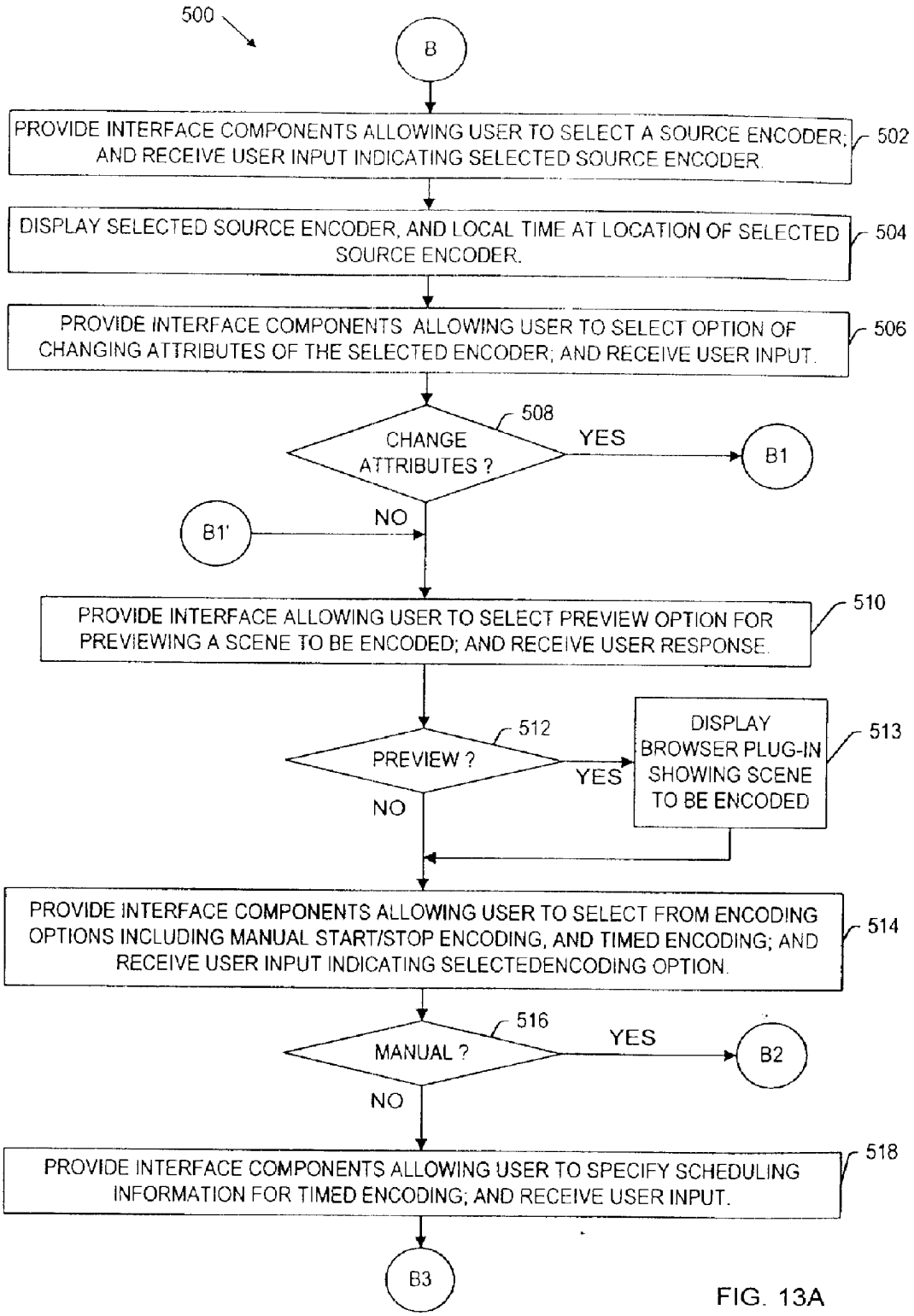

500
B
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT A SOURCE ENCODER; AND RECEIVE USER INPUT INDICATING SELECTED SOURCE ENCODER.
502
DISPLAY SELECTED SOURCE ENCODER, AND LOCAL TIME AT LOCATION OF SELECTED SOURCE ENCODER.
504
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT OPTION OF CHANGING ATTRIBUTES OF THE SELECTED ENCODER; AND RECEIVE USER INPUT.
506
508
CHANGE ATTRIBUTES ?
YES
B1
B1'
NO
PROVIDE INTERFACE ALLOWING USER TO SELECT PREVIEW OPTION FOR PREVIEWING A SCENE TO BE ENCODED; AND RECEIVE USER RESPONSE.
510
512
PREVIEW ?
YES
DISPLAY BROWSER PLUG-IN SHOWING SCENE TO BE ENCODED
513
NO
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT FROM ENCODING OPTIONS INCLUDING MANUAL START/STOP ENCODING, AND TIMED ENCODING; AND RECEIVE USER INPUT INDICATING SELECTEDENCODING OPTION.
514
516
MANUAL ?
YES
B2
NO
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SPECIFY SCHEDULING INFORMATION FOR TIMED ENCODING; AND RECEIVE USER INPUT.
518
B3

FIG. 13A

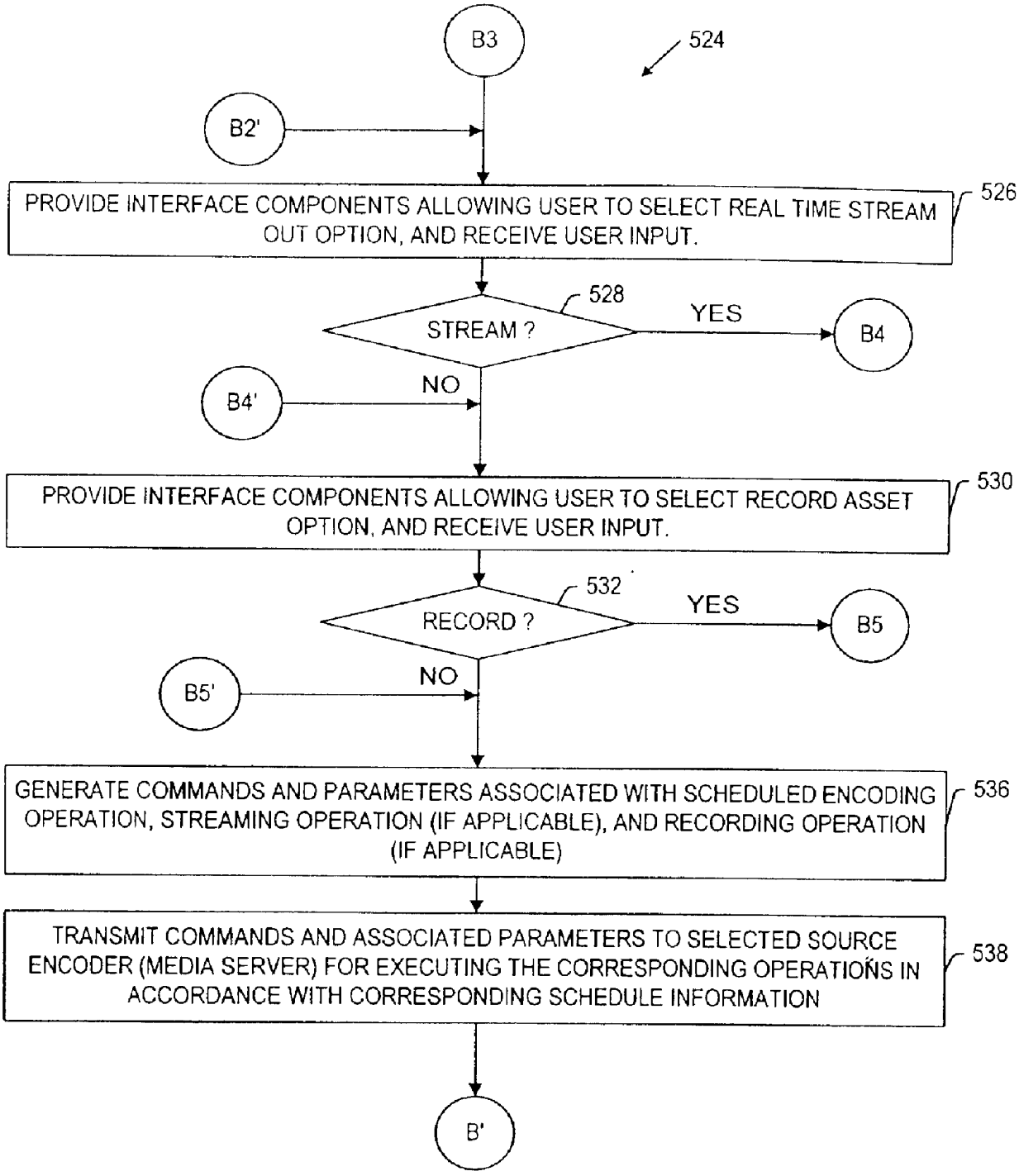

B3
524
B2'
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT REAL TIME STREAM OUT OPTION, AND RECEIVE USER INPUT.
526
528
STREAM ?
YES
B4
B4'
NO
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT RECORD ASSET OPTION, AND RECEIVE USER INPUT.
530
532
RECORD ?
YES
B5
B5'
NO
GENERATE COMMANDS AND PARAMETERS ASSOCIATED WITH SCHEDULED ENCODING OPERATION, STREAMING OPERATION (IF APPLICABLE), AND RECORDING OPERATION (IF APPLICABLE)
536
TRANSMIT COMMANDS AND ASSOCIATED PARAMETERS TO SELECTED SOURCE ENCODER (MEDIA SERVER) FOR EXECUTING THE CORRESPONDING OPERATIONS IN ACCORDANCE WITH CORRESPONDING SCHEDULE INFORMATION
538
B'

FIG. 13B

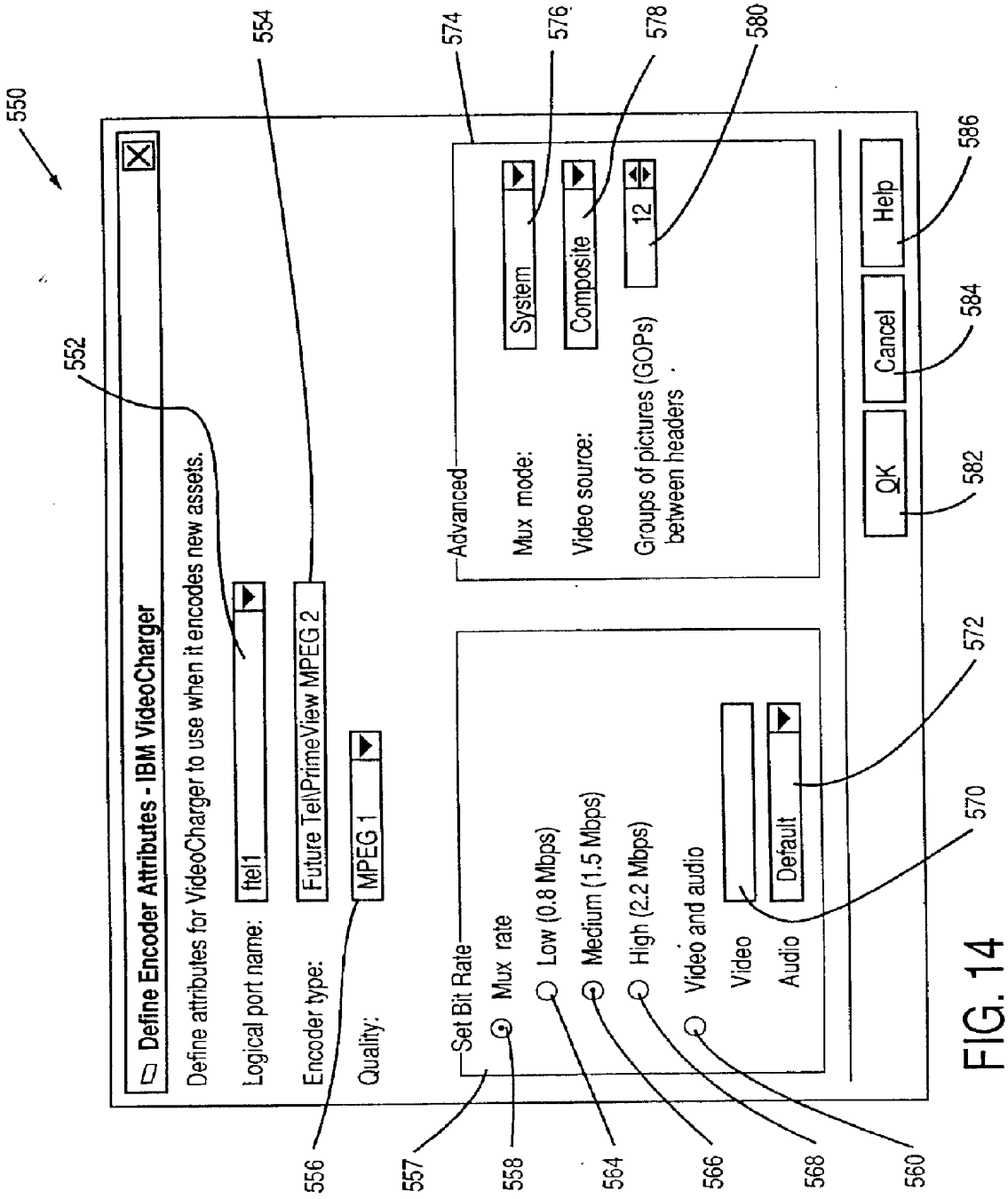

550
552
554
556
557
558
560
564
566
568
570
572
574
576
578
580
582
584
586
Define Encoder Attributes - IBM VideoCharger
Define attributes for VideoCharger to use when it encodes new assets.
Logical port name:
ftel1
Encoder type:
Future Tel\PrimeView MPEG 2
Quality:
MPEG 1
Set Bit Rate
Mux rate
Low (0.8 Mbps)
Medium (1.5 Mbps)
High (2.2 Mbps)
Video and audio
Video
Audio
Default
Advanced
Mux mode:
System
Video source:
Composite
Groups of pictures (GOPs) between headers
12
OK
Cancel
Help

FIG. 14

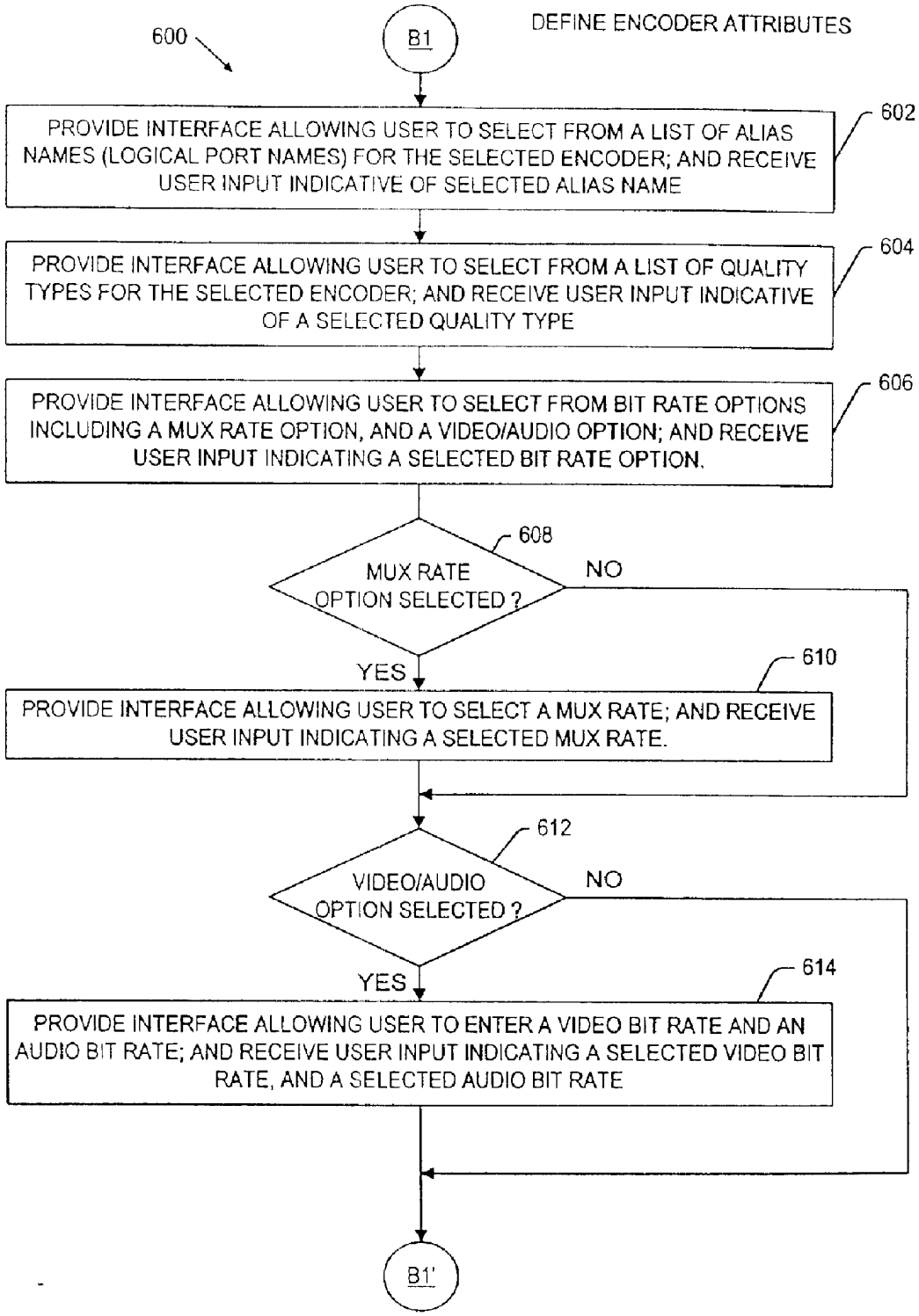

600
B1
DEFINE ENCODER ATTRIBUTES
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A LIST OF ALIAS NAMES (LOGICAL PORT NAMES) FOR THE SELECTED ENCODER; AND RECEIVE USER INPUT INDICATIVE OF SELECTED ALIAS NAME
602
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A LIST OF QUALITY TYPES FOR THE SELECTED ENCODER; AND RECEIVE USER INPUT INDICATIVE OF A SELECTED QUALITY TYPE
604
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM BIT RATE OPTIONS INCLUDING A MUX RATE OPTION, AND A VIDEO/AUDIO OPTION; AND RECEIVE USER INPUT INDICATING A SELECTED BIT RATE OPTION.
606
608
MUX RATE OPTION SELECTED ?
NO
YES
610
PROVIDE INTERFACE ALLOWING USER TO SELECT A MUX RATE; AND RECEIVE USER INPUT INDICATING A SELECTED MUX RATE.
612
VIDEO/AUDIO OPTION SELECTED ?
NO
YES
614
PROVIDE INTERFACE ALLOWING USER TO ENTER A VIDEO BIT RATE AND AN AUDIO BIT RATE; AND RECEIVE USER INPUT INDICATING A SELECTED VIDEO BIT RATE, AND A SELECTED AUDIO BIT RATE
B1'

FIG. 15

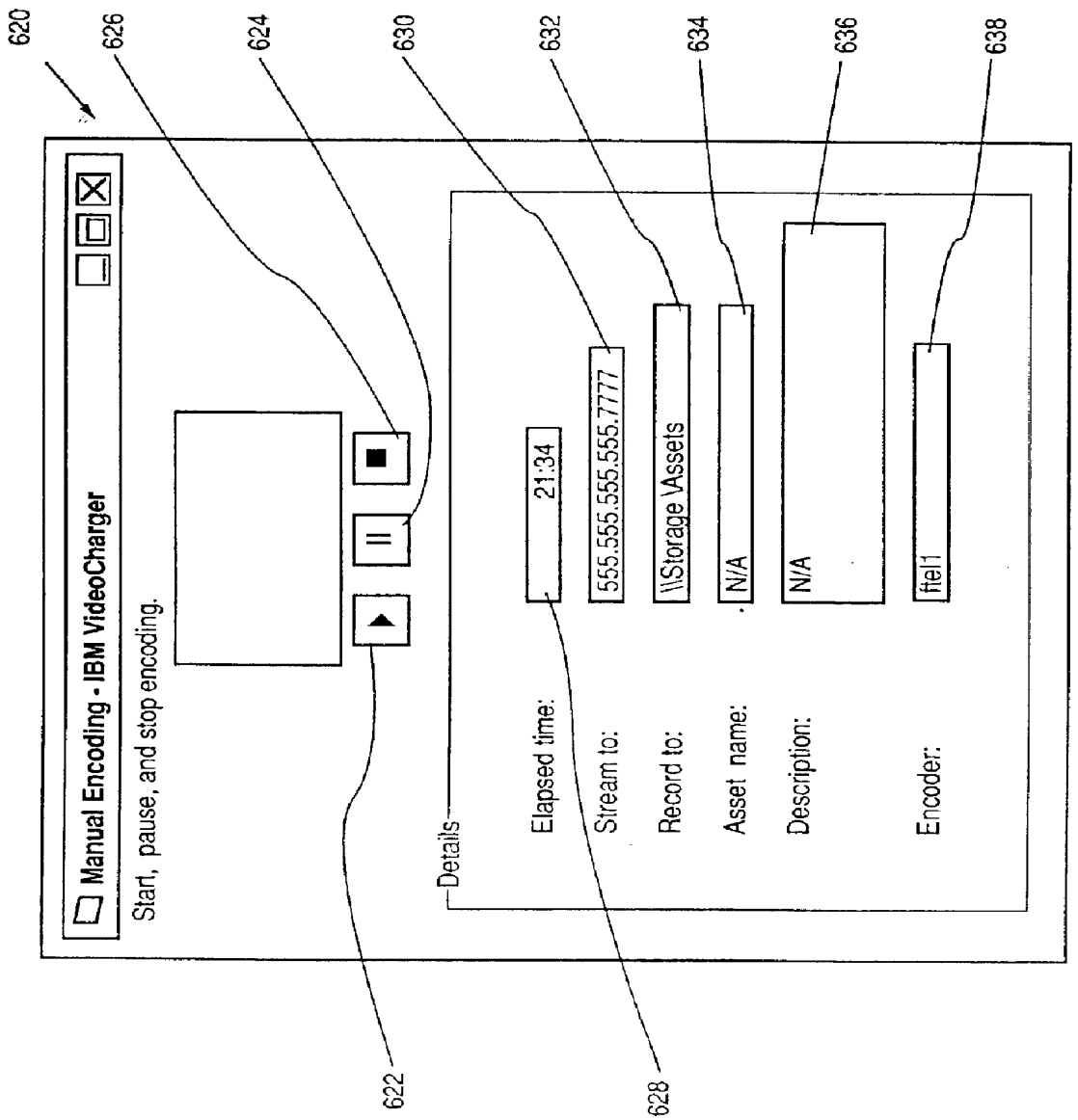

620
626
624
630
632
634
636
638
Manual Encoding - IBM VideoCharger
Start, pause, and stop encoding.
Details
Elapsed time:
21:34
Stream to:
555.555.555.555.7777
Record to:
\\Storage \Assets
Asset name:
N/A
Description:
N/A
Encoder:
file1
622
628

FIG. 16

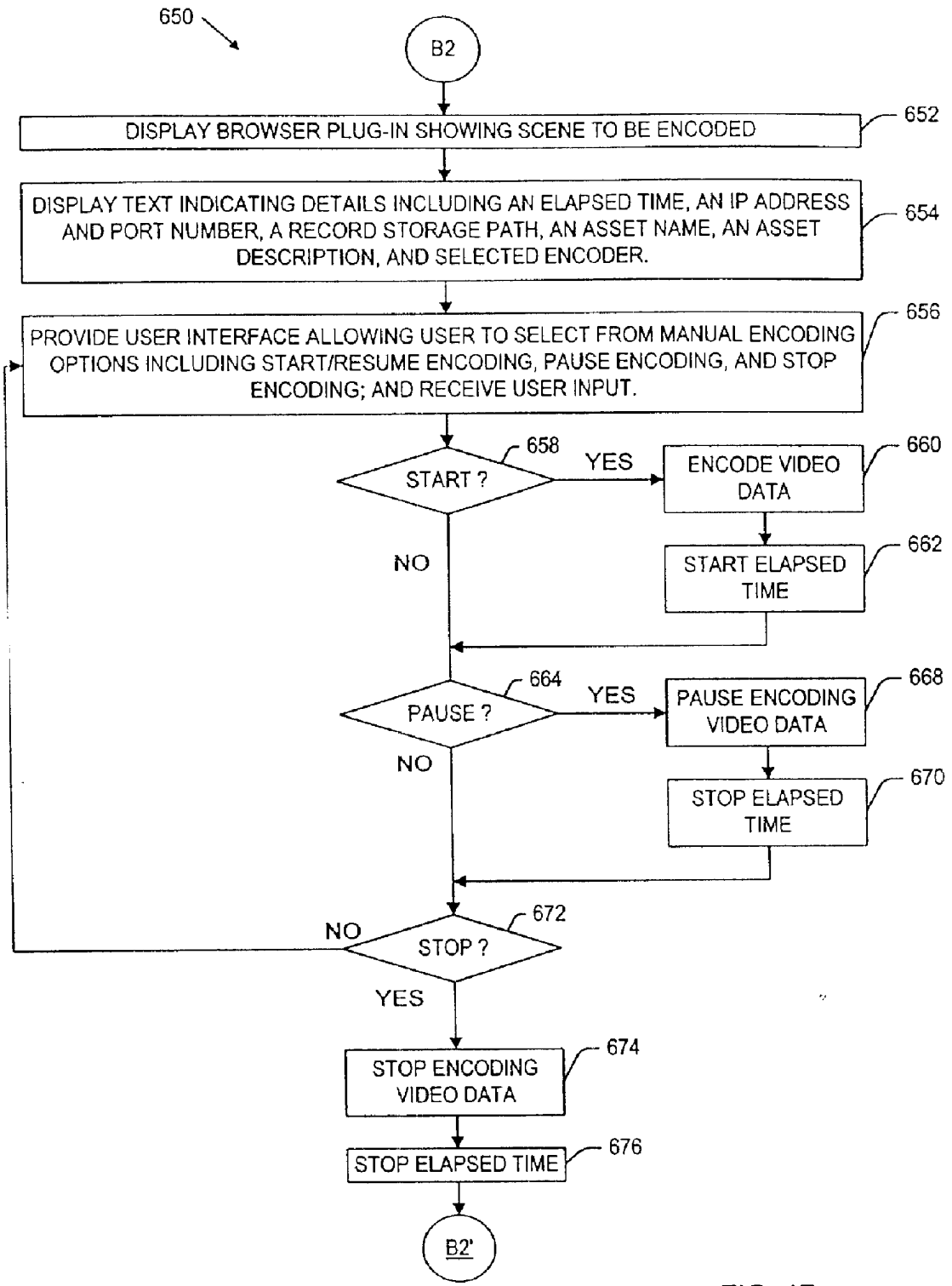

650
B2
DISPLAY BROWSER PLUG-IN SHOWING SCENE TO BE ENCODED
652
DISPLAY TEXT INDICATING DETAILS INCLUDING AN ELAPSED TIME, AN IP ADDRESS AND PORT NUMBER, A RECORD STORAGE PATH, AN ASSET NAME, AN ASSET DESCRIPTION, AND SELECTED ENCODER.
654
PROVIDE USER INTERFACE ALLOWING USER TO SELECT FROM MANUAL ENCODING OPTIONS INCLUDING START/RESUME ENCODING, PAUSE ENCODING, AND STOP ENCODING; AND RECEIVE USER INPUT.
656
START ?
658
YES
ENCODE VIDEO DATA
660
NO
START ELAPSED TIME
662
PAUSE ?
664
YES
PAUSE ENCODING VIDEO DATA
668
NO
STOP ELAPSED TIME
670
NO
STOP ?
672
YES
STOP ENCODING VIDEO DATA
674
STOP ELAPSED TIME
676
B2'

FIG. 17

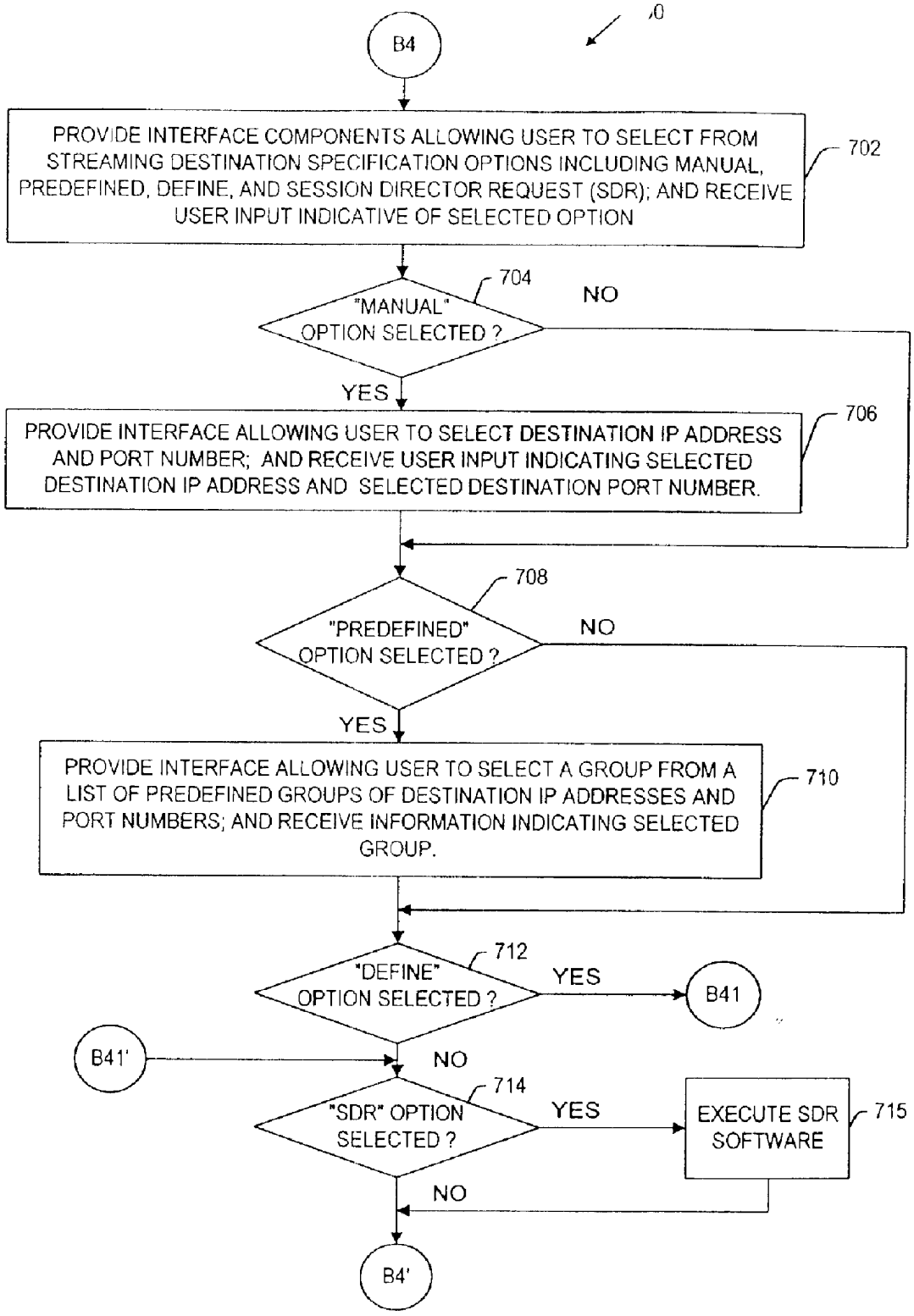

700
B4
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT FROM STREAMING DESTINATION SPECIFICATION OPTIONS INCLUDING MANUAL, PREDEFINED, DEFINE, AND SESSION DIRECTOR REQUEST (SDR); AND RECEIVE USER INPUT INDICATIVE OF SELECTED OPTION
702
"MANUAL" OPTION SELECTED ?
704
NO
YES
PROVIDE INTERFACE ALLOWING USER TO SELECT DESTINATION IP ADDRESS AND PORT NUMBER; AND RECEIVE USER INPUT INDICATING SELECTED DESTINATION IP ADDRESS AND SELECTED DESTINATION PORT NUMBER.
706
"PREDEFINED" OPTION SELECTED ?
708
NO
YES
PROVIDE INTERFACE ALLOWING USER TO SELECT A GROUP FROM A LIST OF PREDEFINED GROUPS OF DESTINATION IP ADDRESSES AND PORT NUMBERS; AND RECEIVE INFORMATION INDICATING SELECTED GROUP.
710
"DEFINE" OPTION SELECTED ?
712
YES
B41
B41'
NO
"SDR" OPTION SELECTED ?
714
YES
EXECUTE SDR SOFTWARE
715
NO
B4'

FIG. 18

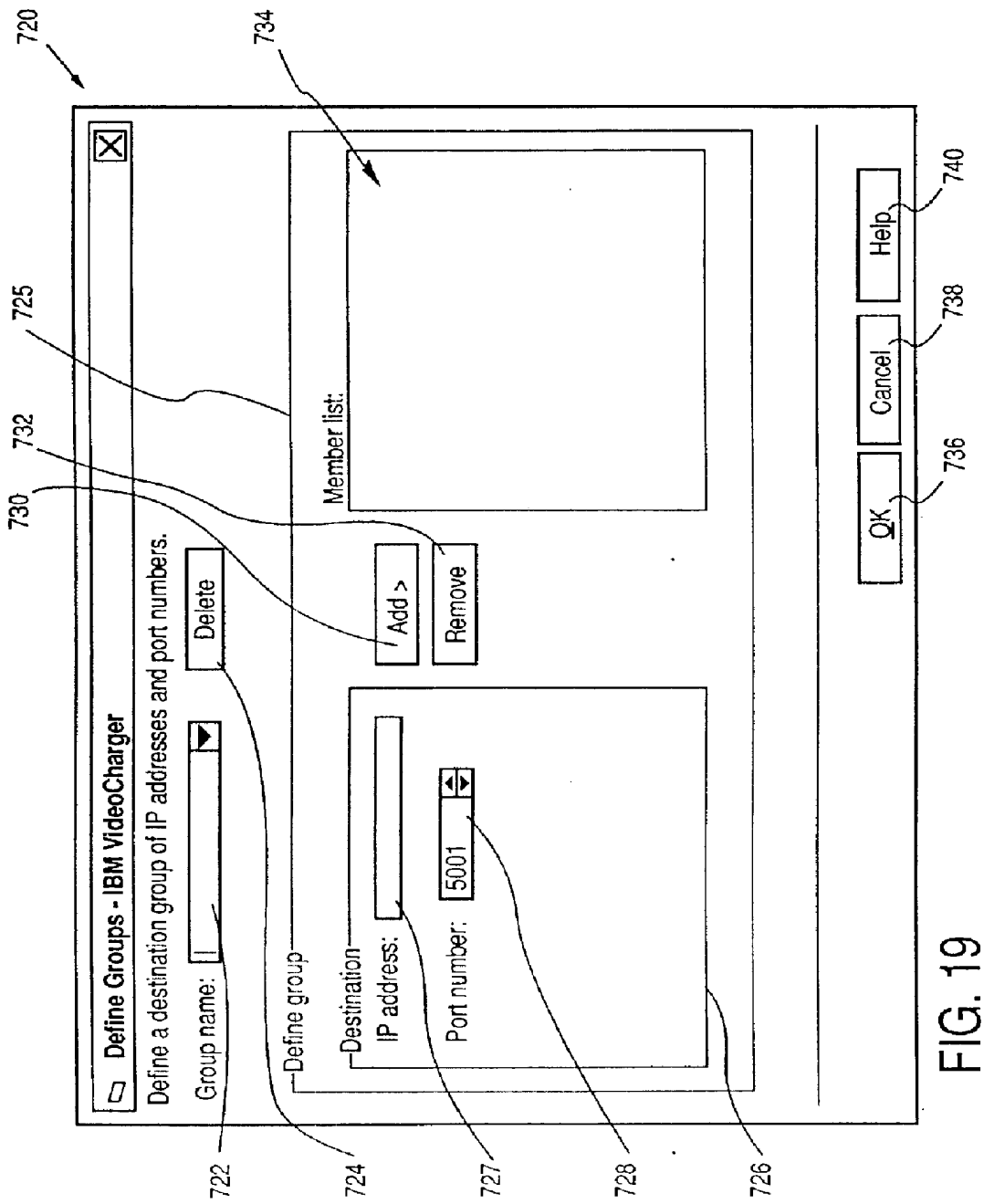

720
734
725
732
730
740
738
736
Define Groups - IBM VideoCharger
Define a destination group of IP addresses and port numbers.
Group name:
Delete
Define group
Destination
IP address:
Port number:
5001
Add >
Remove
Member list:
Help
Cancel
OK
722
724
727
728
726

FIG. 19

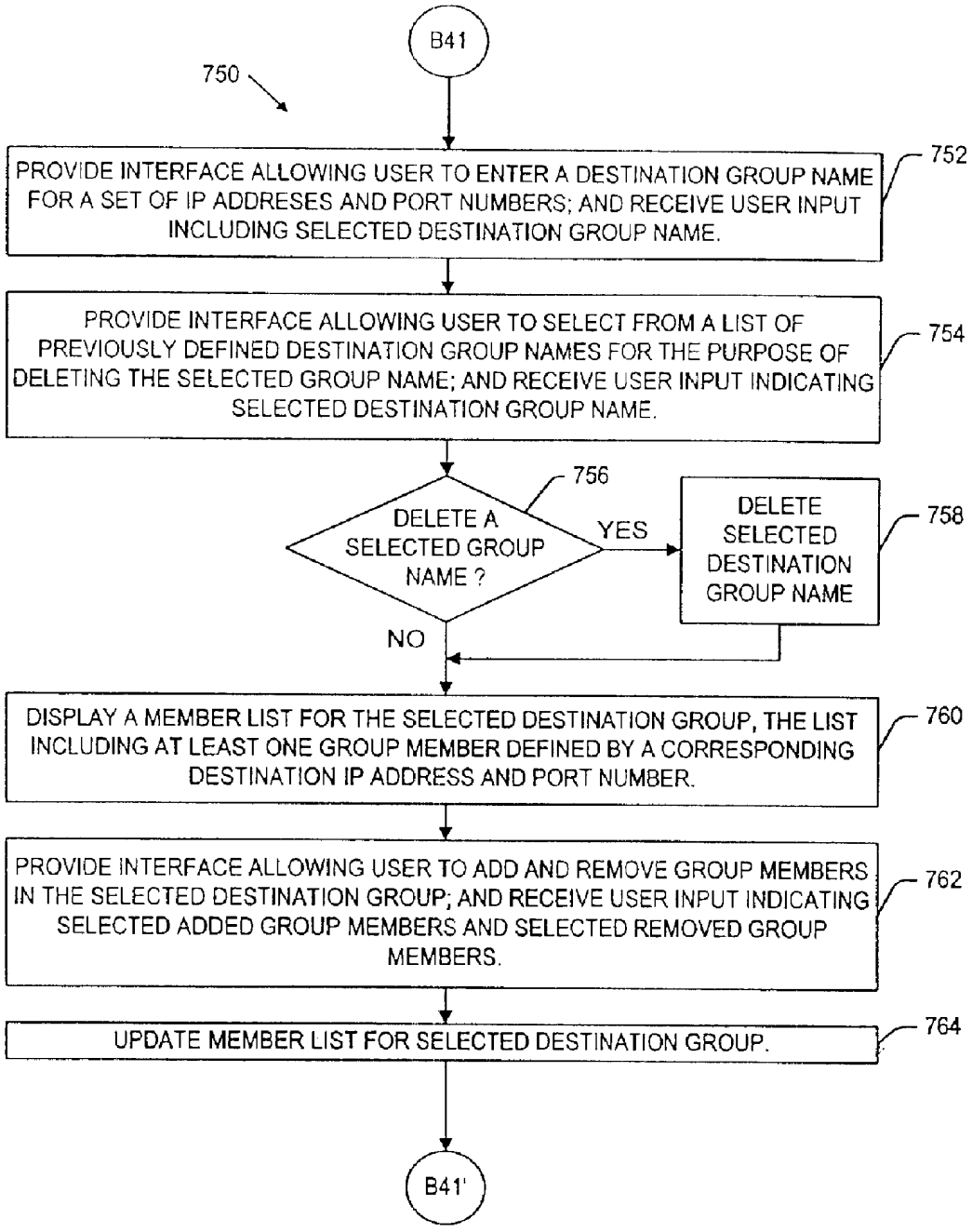

B41
750
PROVIDE INTERFACE ALLOWING USER TO ENTER A DESTINATION GROUP NAME FOR A SET OF IP ADDRESES AND PORT NUMBERS; AND RECEIVE USER INPUT INCLUDING SELECTED DESTINATION GROUP NAME.
752
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A LIST OF PREVIOUSLY DEFINED DESTINATION GROUP NAMES FOR THE PURPOSE OF DELETING THE SELECTED GROUP NAME; AND RECEIVE USER INPUT INDICATING SELECTED DESTINATION GROUP NAME.
754
756
DELETE A SELECTED GROUP NAME ?
YES
DELETE SELECTED DESTINATION GROUP NAME
758
NO
DISPLAY A MEMBER LIST FOR THE SELECTED DESTINATION GROUP, THE LIST INCLUDING AT LEAST ONE GROUP MEMBER DEFINED BY A CORRESPONDING DESTINATION IP ADDRESS AND PORT NUMBER.
760
PROVIDE INTERFACE ALLOWING USER TO ADD AND REMOVE GROUP MEMBERS IN THE SELECTED DESTINATION GROUP; AND RECEIVE USER INPUT INDICATING SELECTED ADDED GROUP MEMBERS AND SELECTED REMOVED GROUP MEMBERS.
762
UPDATE MEMBER LIST FOR SELECTED DESTINATION GROUP.
764
B41'

FIG. 20

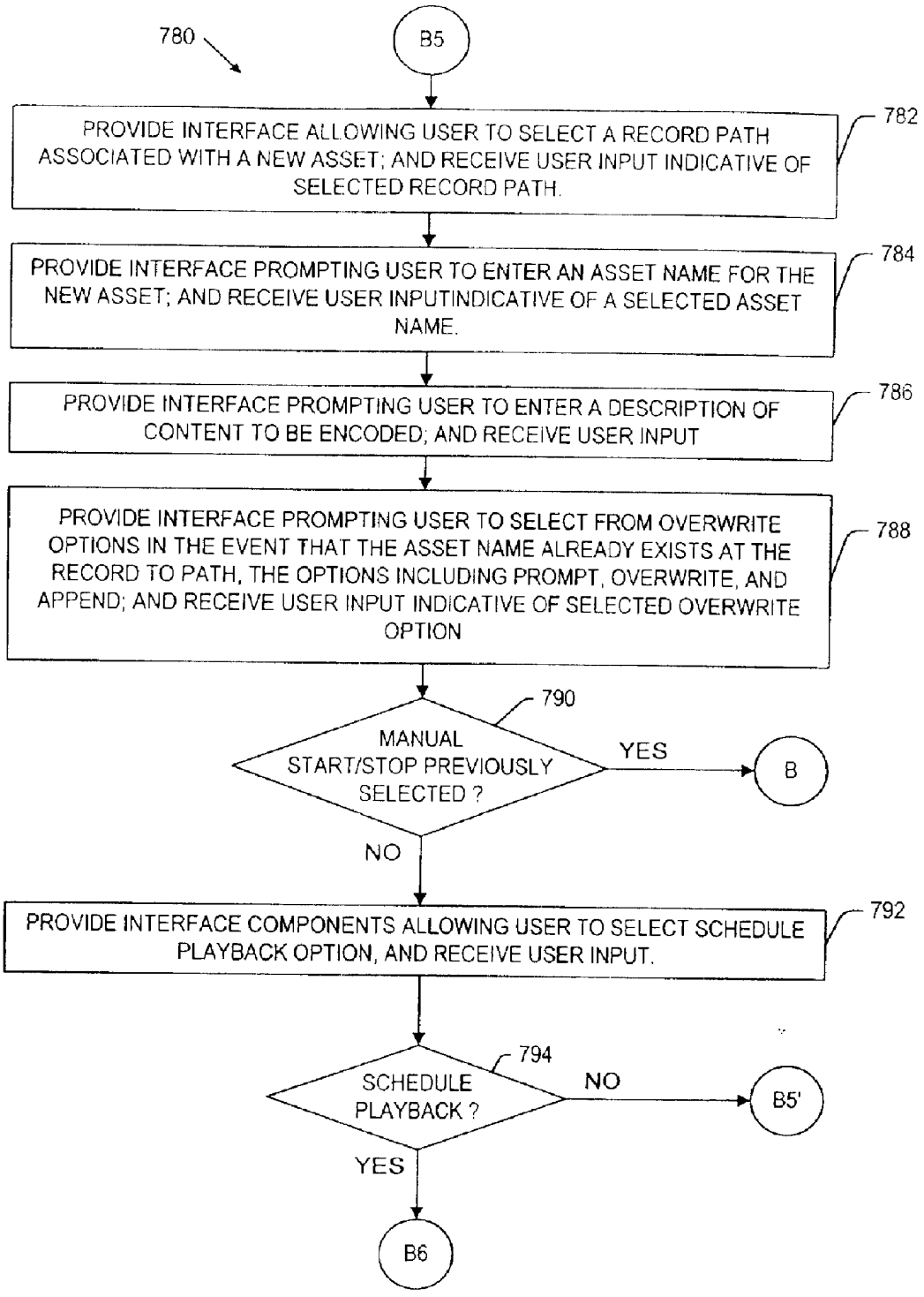

780
B5
PROVIDE INTERFACE ALLOWING USER TO SELECT A RECORD PATH ASSOCIATED WITH A NEW ASSET; AND RECEIVE USER INPUT INDICATIVE OF SELECTED RECORD PATH.
782
PROVIDE INTERFACE PROMPTING USER TO ENTER AN ASSET NAME FOR THE NEW ASSET; AND RECEIVE USER INPUTINDICATIVE OF A SELECTED ASSET NAME.
784
PROVIDE INTERFACE PROMPTING USER TO ENTER A DESCRIPTION OF CONTENT TO BE ENCODED; AND RECEIVE USER INPUT
786
PROVIDE INTERFACE PROMPTING USER TO SELECT FROM OVERWRITE OPTIONS IN THE EVENT THAT THE ASSET NAME ALREADY EXISTS AT THE RECORD TO PATH, THE OPTIONS INCLUDING PROMPT, OVERWRITE, AND APPEND; AND RECEIVE USER INPUT INDICATIVE OF SELECTED OVERWRITE OPTION
788
790
MANUAL START/STOP PREVIOUSLY SELECTED ?
YES
B
NO
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT SCHEDULE PLAYBACK OPTION, AND RECEIVE USER INPUT.
792
794
SCHEDULE PLAYBACK ?
NO
B5'
YES
B6

FIG. 21

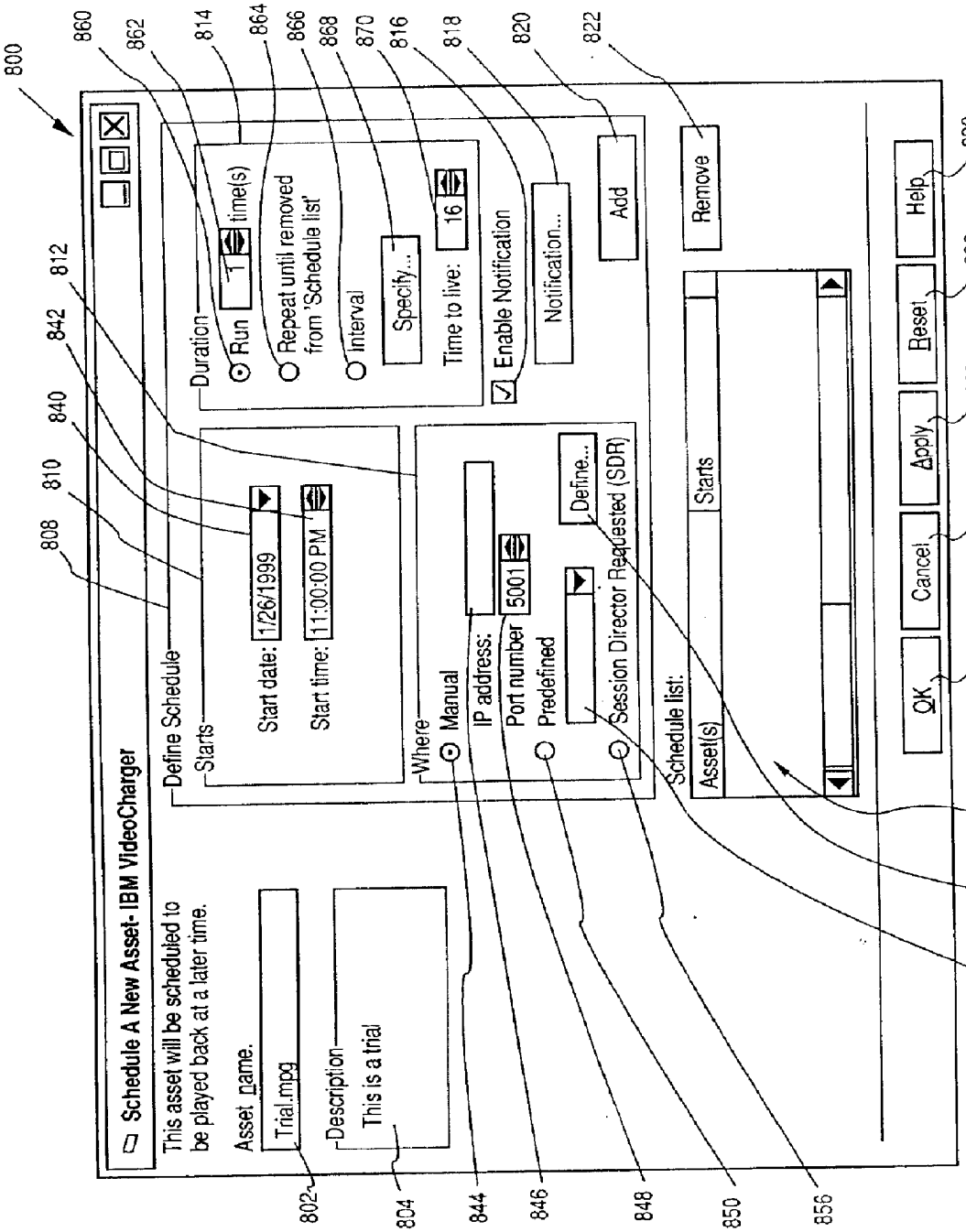

800
Schedule A New Asset- IBM VideoCharger
This asset will be scheduled to be played back at a later time.
Asset name.
Trial.mpg
Description
This is a trial
Define Schedule
Starts
Start date: 1/26/1999
Start time: 11:00:00 PM
Where
Manual
IP address:
Port number 5001
Predefined
Define...
Session Director Requested (SDR)
Duration
Run 1 time(s)
Repeat until removed from 'Schedule list'
Interval
Specify...
Time to live: 16
Enable Notification
Notification...
Add
Remove
Schedule list:
Asset(s)
Starts
OK
Cancel
Apply
Reset
Help
802
804
808
810
812
814
816
818
820
822
824
826
828
830
832
840
842
844
846
848
850
852
854
856
860
862
864
866
868
870

FIG. 22

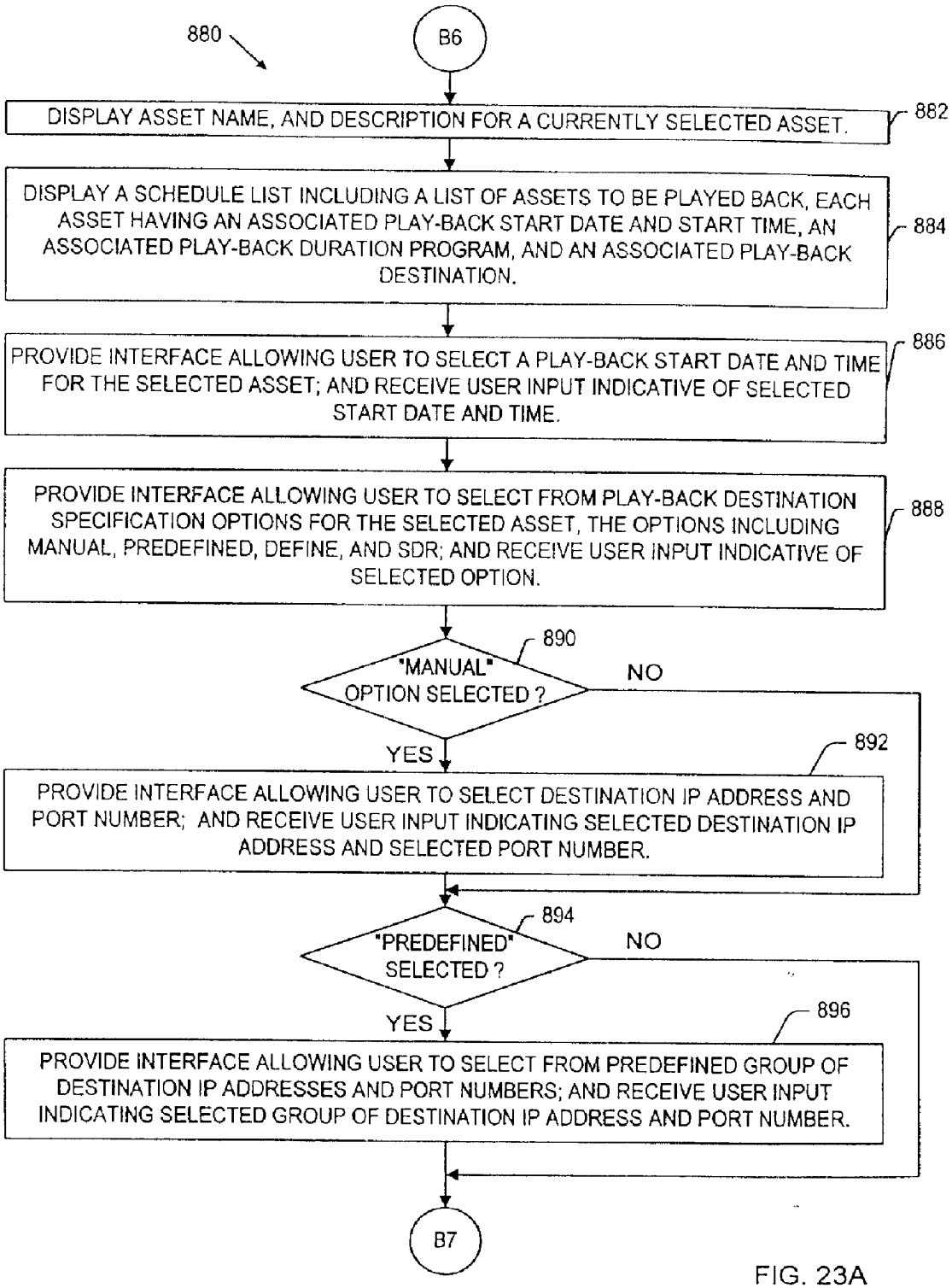

880
B6
DISPLAY ASSET NAME, AND DESCRIPTION FOR A CURRENTLY SELECTED ASSET.
882
DISPLAY A SCHEDULE LIST INCLUDING A LIST OF ASSETS TO BE PLAYED BACK, EACH ASSET HAVING AN ASSOCIATED PLAY-BACK START DATE AND START TIME, AN ASSOCIATED PLAY-BACK DURATION PROGRAM, AND AN ASSOCIATED PLAY-BACK DESTINATION.
884
PROVIDE INTERFACE ALLOWING USER TO SELECT A PLAY-BACK START DATE AND TIME FOR THE SELECTED ASSET; AND RECEIVE USER INPUT INDICATIVE OF SELECTED START DATE AND TIME.
886
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM PLAY-BACK DESTINATION SPECIFICATION OPTIONS FOR THE SELECTED ASSET, THE OPTIONS INCLUDING MANUAL, PREDEFINED, DEFINE, AND SDR; AND RECEIVE USER INPUT INDICATIVE OF SELECTED OPTION.
888
890
"MANUAL" OPTION SELECTED ?
NO
YES
892
PROVIDE INTERFACE ALLOWING USER TO SELECT DESTINATION IP ADDRESS AND PORT NUMBER; AND RECEIVE USER INPUT INDICATING SELECTED DESTINATION IP ADDRESS AND SELECTED PORT NUMBER.
894
"PREDEFINED" SELECTED ?
NO
YES
896
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM PREDEFINED GROUP OF DESTINATION IP ADDRESSES AND PORT NUMBERS; AND RECEIVE USER INPUT INDICATING SELECTED GROUP OF DESTINATION IP ADDRESS AND PORT NUMBER.
B7

FIG. 23A

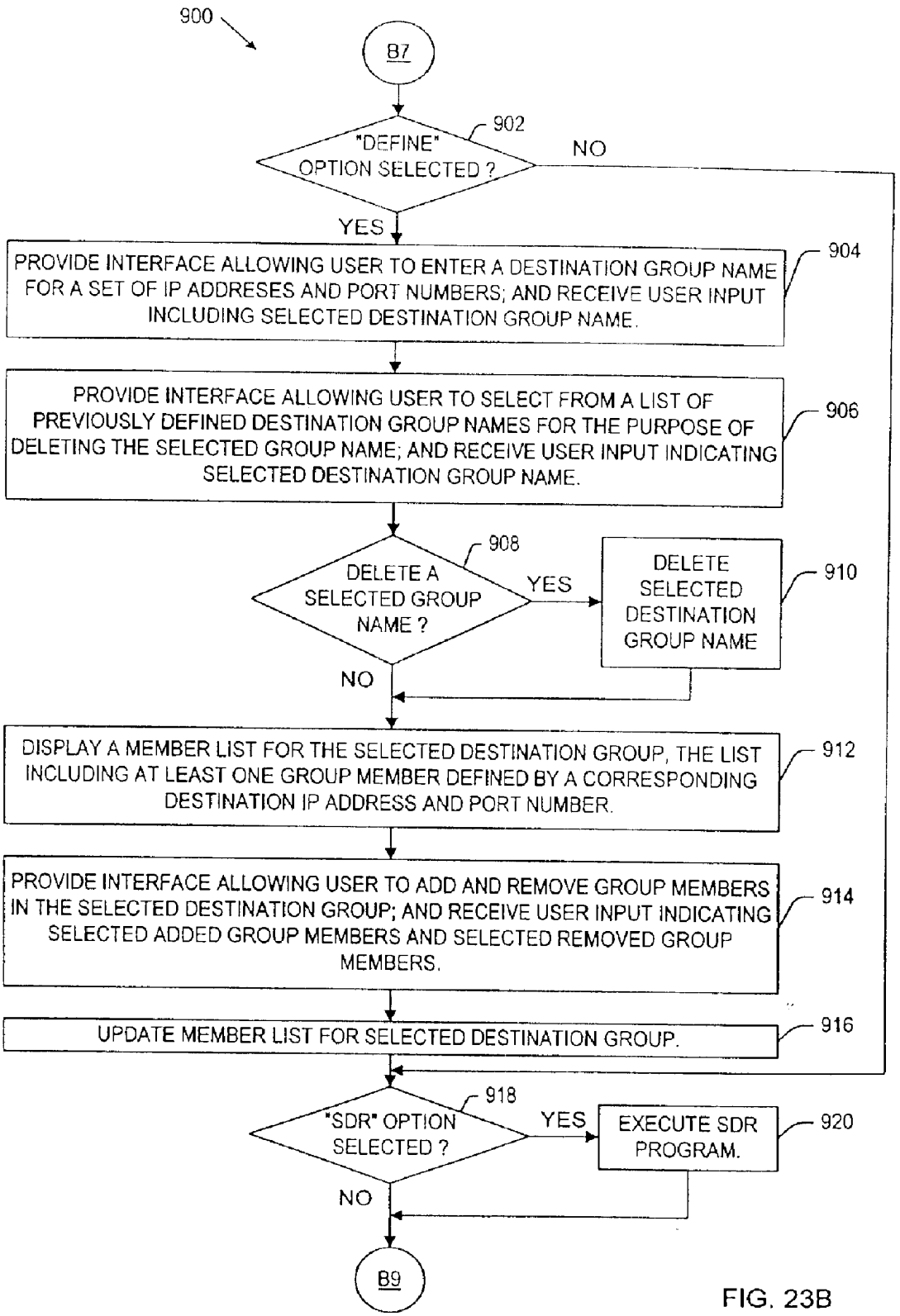

900
B7
902
"DEFINE" OPTION SELECTED ?
NO
YES
PROVIDE INTERFACE ALLOWING USER TO ENTER A DESTINATION GROUP NAME FOR A SET OF IP ADDRESES AND PORT NUMBERS; AND RECEIVE USER INPUT INCLUDING SELECTED DESTINATION GROUP NAME.
904
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A LIST OF PREVIOUSLY DEFINED DESTINATION GROUP NAMES FOR THE PURPOSE OF DELETING THE SELECTED GROUP NAME; AND RECEIVE USER INPUT INDICATING SELECTED DESTINATION GROUP NAME.
906
908
DELETE A SELECTED GROUP NAME ?
YES
DELETE SELECTED DESTINATION GROUP NAME
910
NO
DISPLAY A MEMBER LIST FOR THE SELECTED DESTINATION GROUP, THE LIST INCLUDING AT LEAST ONE GROUP MEMBER DEFINED BY A CORRESPONDING DESTINATION IP ADDRESS AND PORT NUMBER.
912
PROVIDE INTERFACE ALLOWING USER TO ADD AND REMOVE GROUP MEMBERS IN THE SELECTED DESTINATION GROUP; AND RECEIVE USER INPUT INDICATING SELECTED ADDED GROUP MEMBERS AND SELECTED REMOVED GROUP MEMBERS.
914
UPDATE MEMBER LIST FOR SELECTED DESTINATION GROUP.
916
918
"SDR" OPTION SELECTED ?
YES
EXECUTE SDR PROGRAM.
920
NO
B9

FIG. 23B

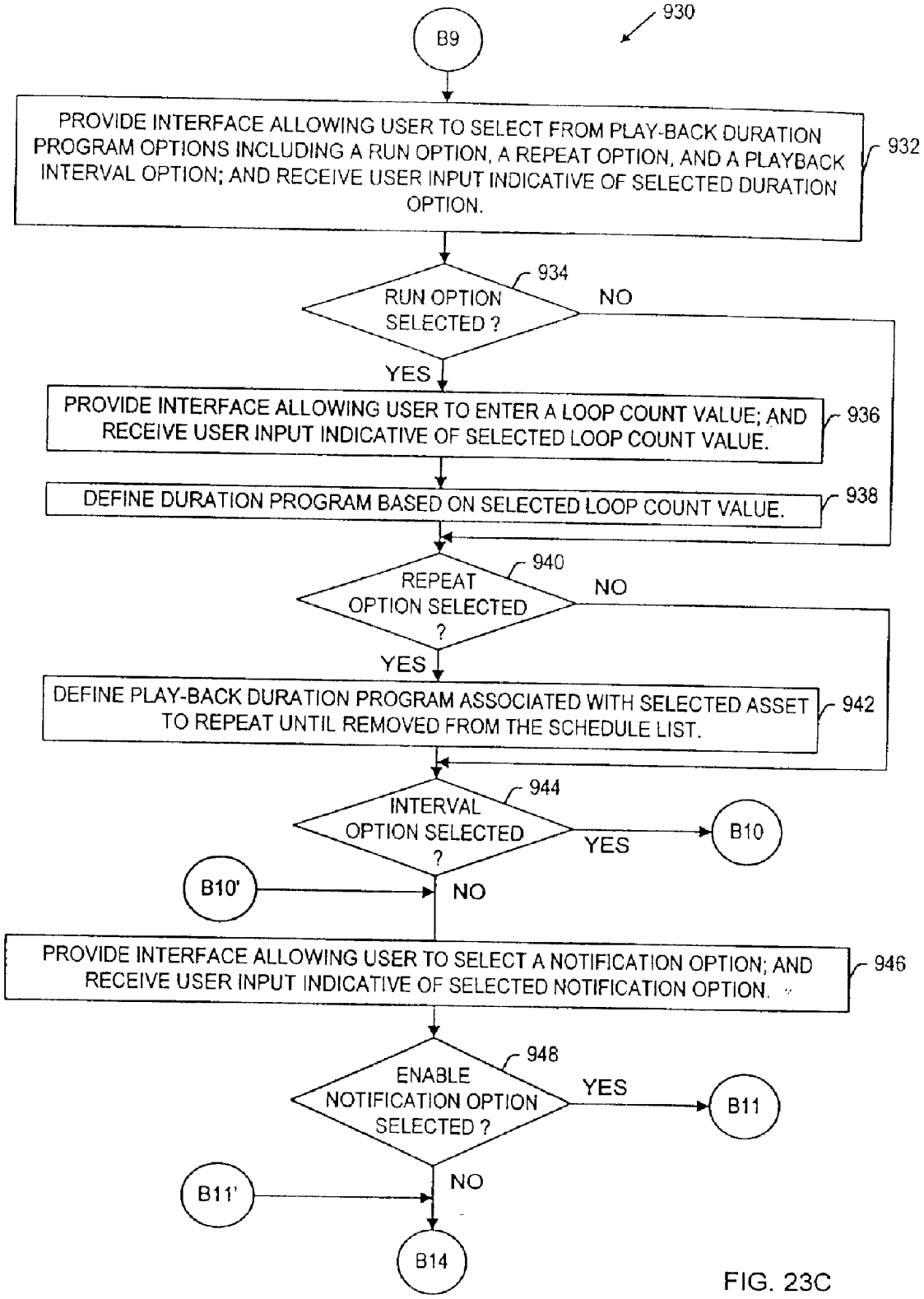

930
B9
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM PLAY-BACK DURATION PROGRAM OPTIONS INCLUDING A RUN OPTION, A REPEAT OPTION, AND A PLAYBACK INTERVAL OPTION; AND RECEIVE USER INPUT INDICATIVE OF SELECTED DURATION OPTION.
932
934
RUN OPTION SELECTED ?
NO
YES
PROVIDE INTERFACE ALLOWING USER TO ENTER A LOOP COUNT VALUE; AND RECEIVE USER INPUT INDICATIVE OF SELECTED LOOP COUNT VALUE.
936
DEFINE DURATION PROGRAM BASED ON SELECTED LOOP COUNT VALUE.
938
940
REPEAT OPTION SELECTED ?
NO
YES
DEFINE PLAY-BACK DURATION PROGRAM ASSOCIATED WITH SELECTED ASSET TO REPEAT UNTIL REMOVED FROM THE SCHEDULE LIST.
942
944
INTERVAL OPTION SELECTED ?
YES
B10
B10'
NO
PROVIDE INTERFACE ALLOWING USER TO SELECT A NOTIFICATION OPTION; AND RECEIVE USER INPUT INDICATIVE OF SELECTED NOTIFICATION OPTION.
946
948
ENABLE NOTIFICATION OPTION SELECTED ?
YES
B11
B11'
NO
B14

FIG. 23C

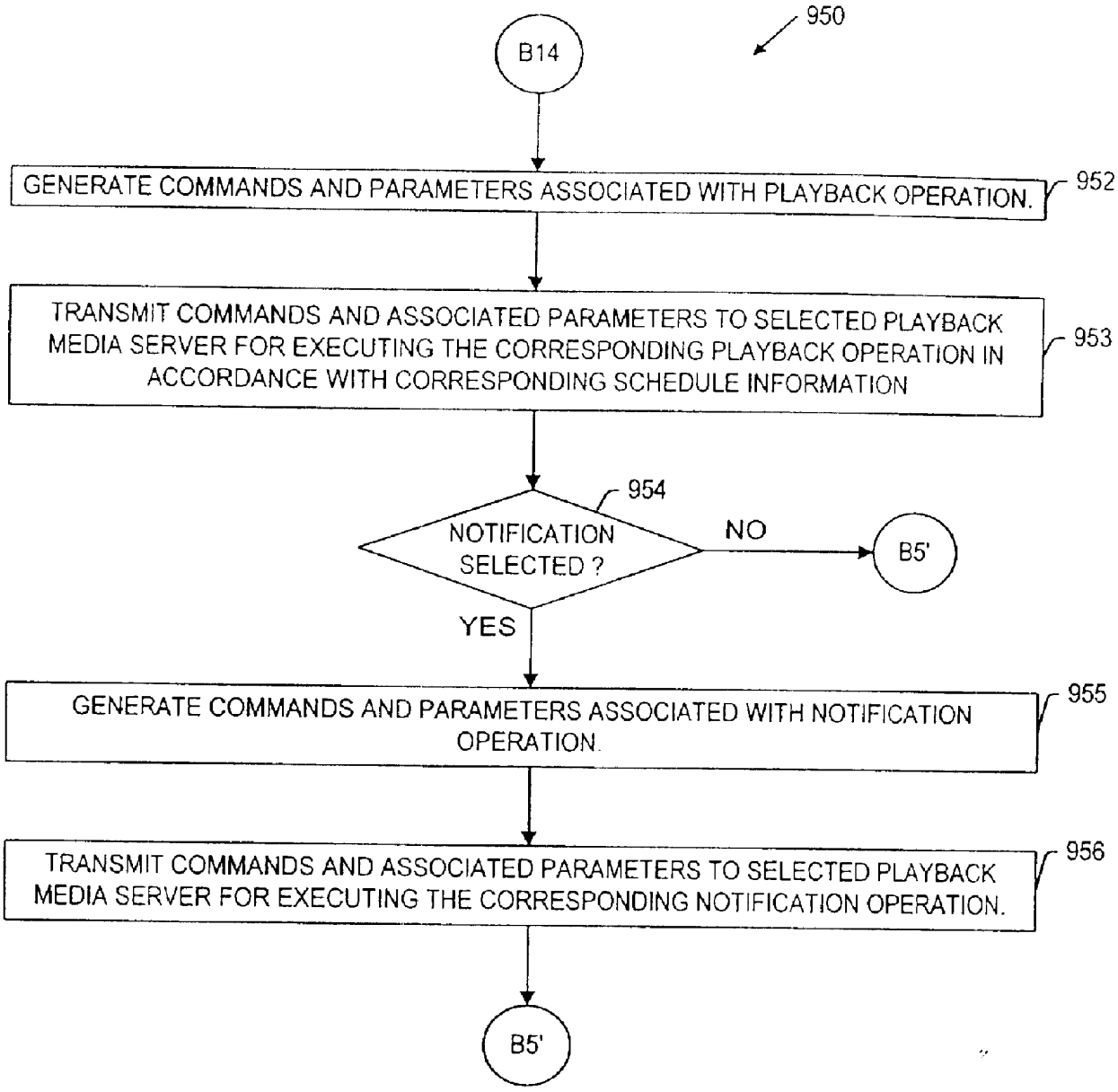

950
B14
GENERATE COMMANDS AND PARAMETERS ASSOCIATED WITH PLAYBACK OPERATION.
952
TRANSMIT COMMANDS AND ASSOCIATED PARAMETERS TO SELECTED PLAYBACK MEDIA SERVER FOR EXECUTING THE CORRESPONDING PLAYBACK OPERATION IN ACCORDANCE WITH CORRESPONDING SCHEDULE INFORMATION
953
954
NOTIFICATION SELECTED ?
NO
B5'
YES
GENERATE COMMANDS AND PARAMETERS ASSOCIATED WITH NOTIFICATION OPERATION.
955
TRANSMIT COMMANDS AND ASSOCIATED PARAMETERS TO SELECTED PLAYBACK MEDIA SERVER FOR EXECUTING THE CORRESPONDING NOTIFICATION OPERATION.
956
B5'

FIG. 23D

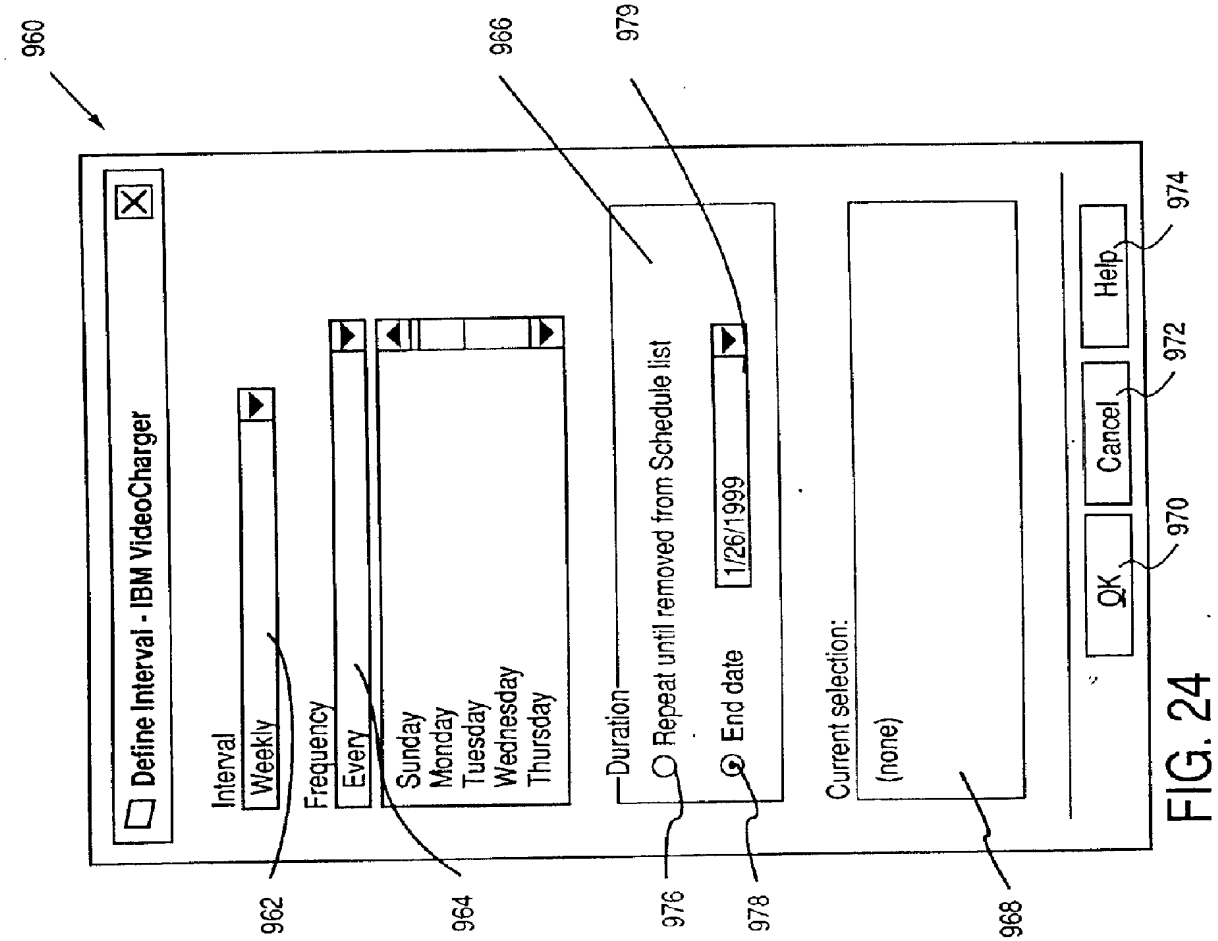

960
966
979
Define Interval - IBM VideoCharger
Interval
Weekly
Frequency
Every
Sunday
Monday
Tuesday
Wednesday
Thursday
Duration
Repeat until removed from Schedule list
End date
1/26/1999
Current selection:
(none)
OK
Cancel
Help
962
964
976
978
968
970
972
974

FIG. 24

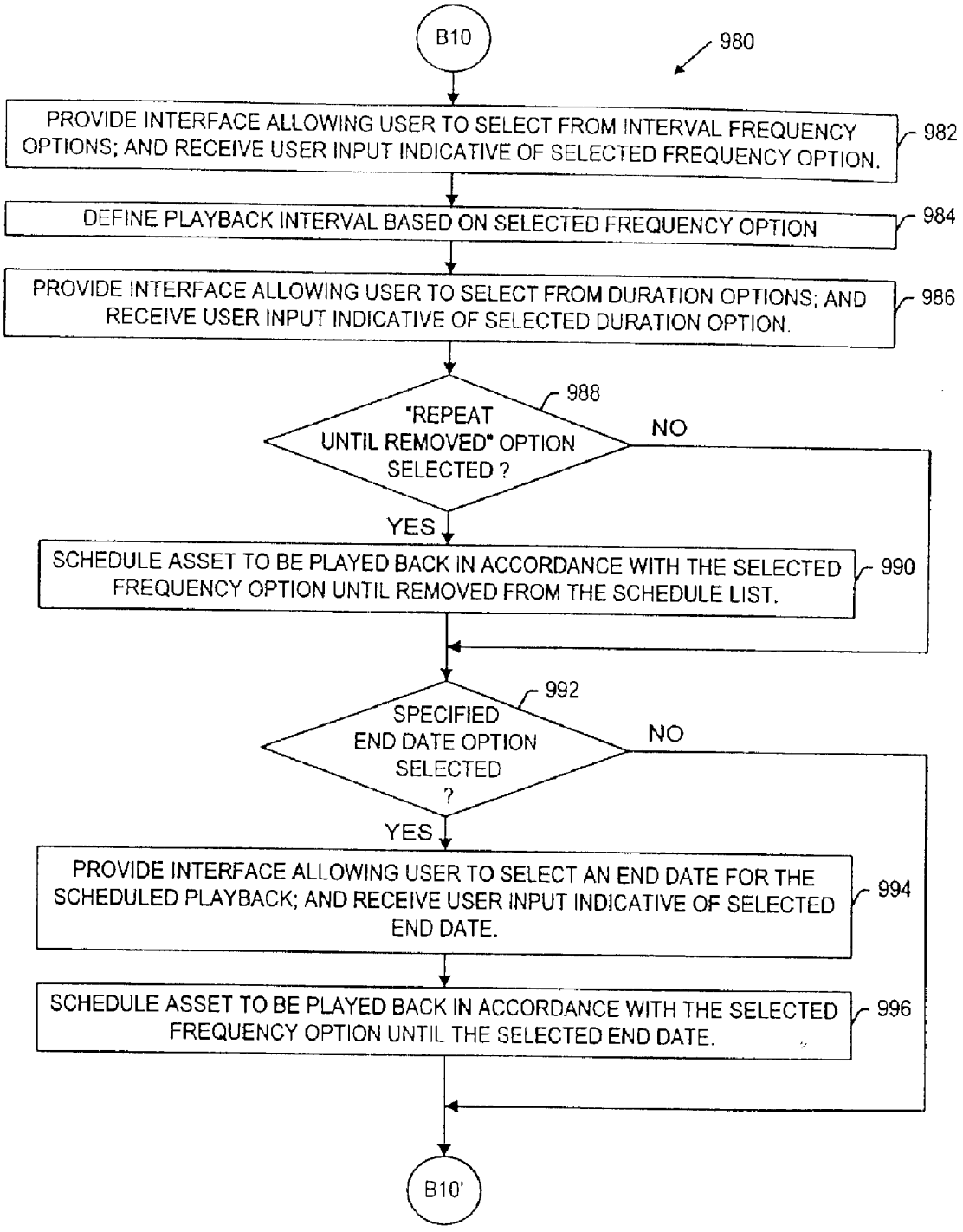

B10
980
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM INTERVAL FREQUENCY OPTIONS; AND RECEIVE USER INPUT INDICATIVE OF SELECTED FREQUENCY OPTION.
982
DEFINE PLAYBACK INTERVAL BASED ON SELECTED FREQUENCY OPTION
984
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM DURATION OPTIONS; AND RECEIVE USER INPUT INDICATIVE OF SELECTED DURATION OPTION.
986
988
"REPEAT UNTIL REMOVED" OPTION SELECTED ?
NO
YES
SCHEDULE ASSET TO BE PLAYED BACK IN ACCORDANCE WITH THE SELECTED FREQUENCY OPTION UNTIL REMOVED FROM THE SCHEDULE LIST.
990
992
SPECIFIED END DATE OPTION SELECTED ?
NO
YES
PROVIDE INTERFACE ALLOWING USER TO SELECT AN END DATE FOR THE SCHEDULED PLAYBACK; AND RECEIVE USER INPUT INDICATIVE OF SELECTED END DATE.
994
SCHEDULE ASSET TO BE PLAYED BACK IN ACCORDANCE WITH THE SELECTED FREQUENCY OPTION UNTIL THE SELECTED END DATE.
996
B10'

FIG. 25

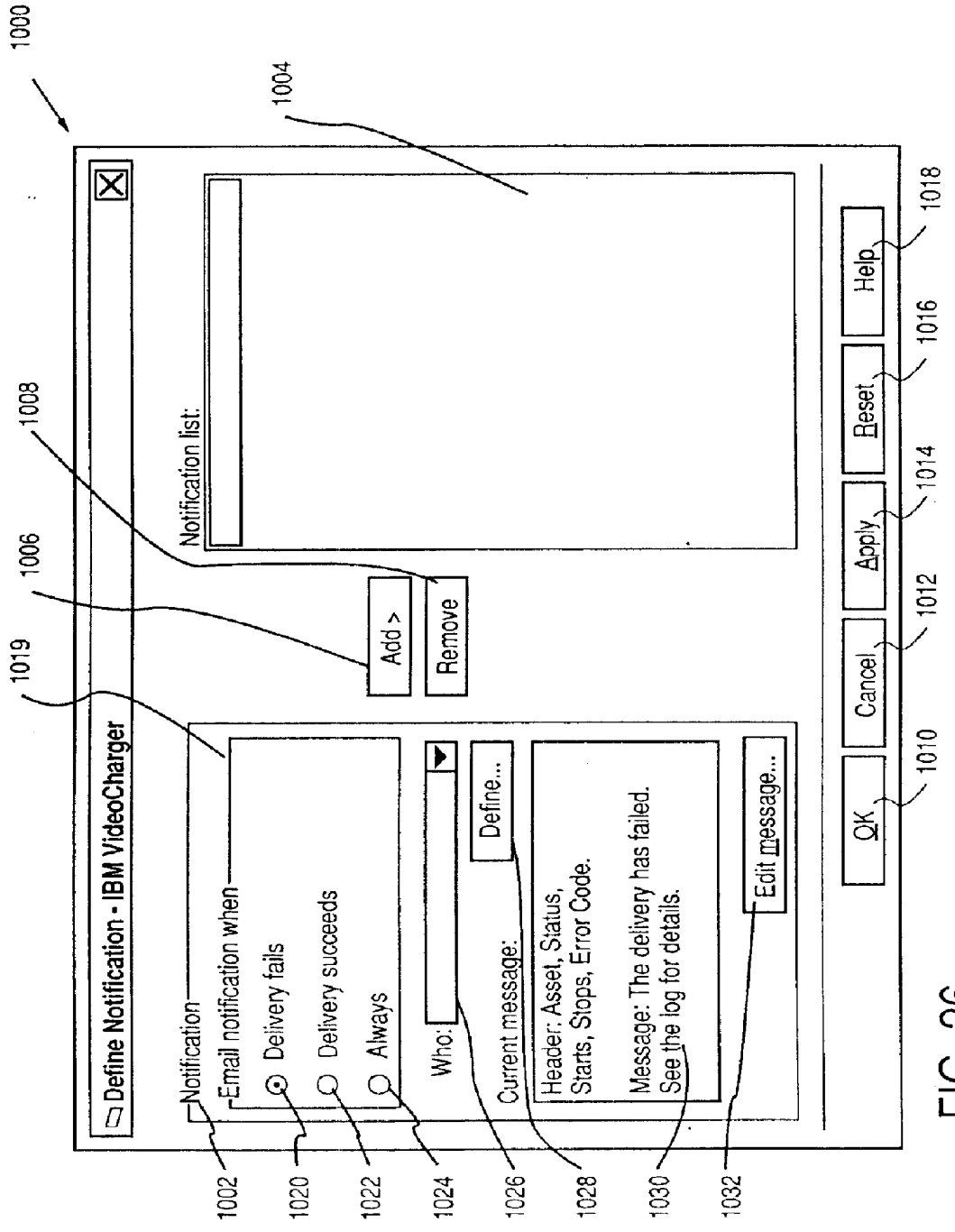

1000
1004
1008
1006
1019
Define Notification - IBM VideoCharger
Notification list:
Add >
Remove
Notification
Email notification when
Delivery fails
Delivery succeeds
Always
Who:
Define...
Current message:
Header: Asset, Status,
Starts, Stops, Error Code.
Message: The delivery has failed.
See the log for details.
Edit message...
OK
Cancel
Apply
Reset
Help
1002
1020
1022
1024
1026
1028
1030
1032
1010
1012
1014
1016
1018

FIG. 26

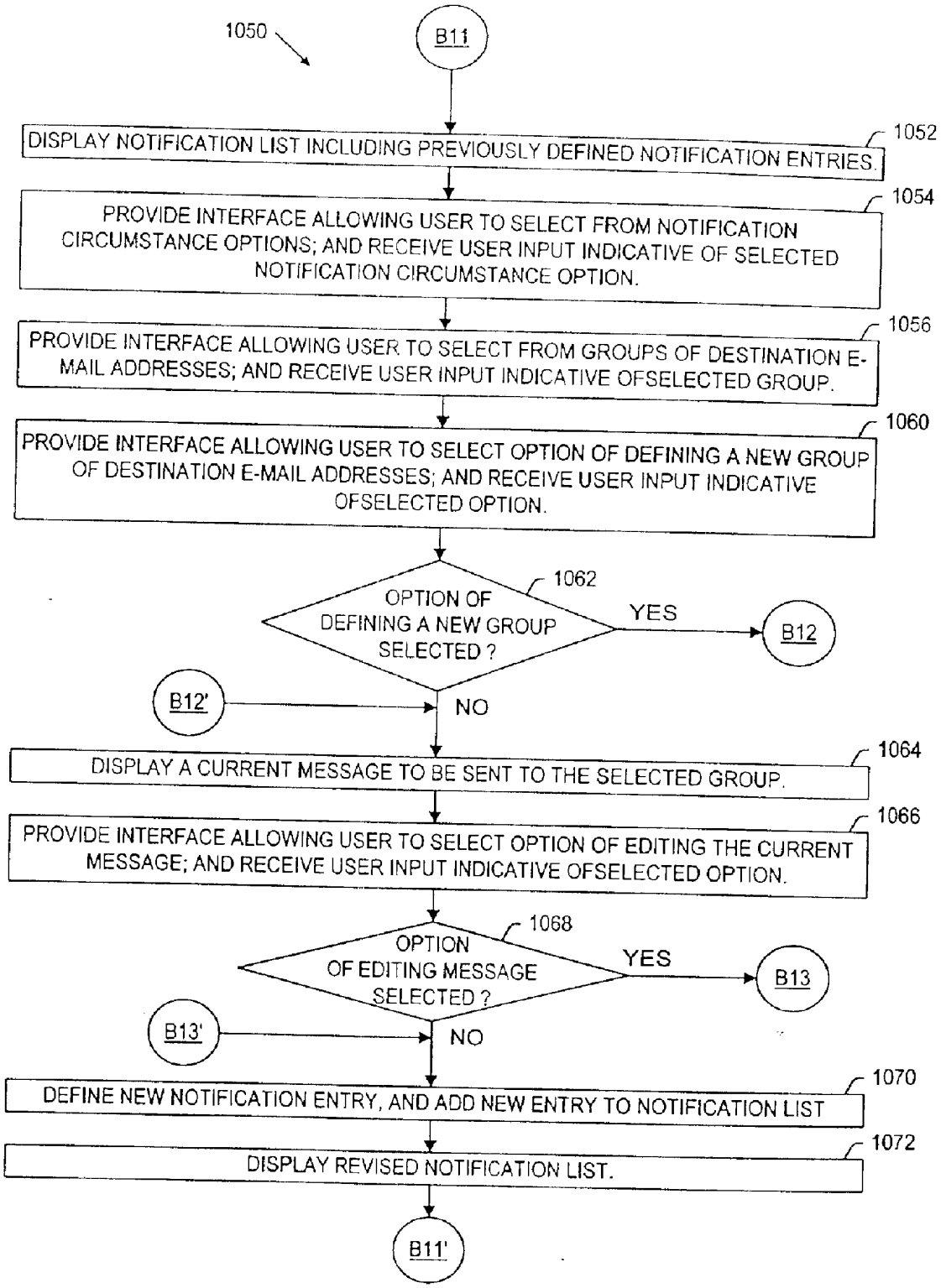

1050
B11
DISPLAY NOTIFICATION LIST INCLUDING PREVIOUSLY DEFINED NOTIFICATION ENTRIES.
1052
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM NOTIFICATION CIRCUMSTANCE OPTIONS; AND RECEIVE USER INPUT INDICATIVE OF SELECTED NOTIFICATION CIRCUMSTANCE OPTION.
1054
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM GROUPS OF DESTINATION E-MAIL ADDRESSES; AND RECEIVE USER INPUT INDICATIVE OFSELECTED GROUP.
1056
PROVIDE INTERFACE ALLOWING USER TO SELECT OPTION OF DEFINING A NEW GROUP OF DESTINATION E-MAIL ADDRESSES; AND RECEIVE USER INPUT INDICATIVE OFSELECTED OPTION.
1060
OPTION OF DEFINING A NEW GROUP SELECTED ?
1062
YES
B12
B12'
NO
DISPLAY A CURRENT MESSAGE TO BE SENT TO THE SELECTED GROUP.
1064
PROVIDE INTERFACE ALLOWING USER TO SELECT OPTION OF EDITING THE CURRENT MESSAGE; AND RECEIVE USER INPUT INDICATIVE OFSELECTED OPTION.
1066
OPTION OF EDITING MESSAGE SELECTED ?
1068
YES
B13
B13'
NO
DEFINE NEW NOTIFICATION ENTRY, AND ADD NEW ENTRY TO NOTIFICATION LIST
1070
DISPLAY REVISED NOTIFICATION LIST.
1072
B11'

FIG. 27

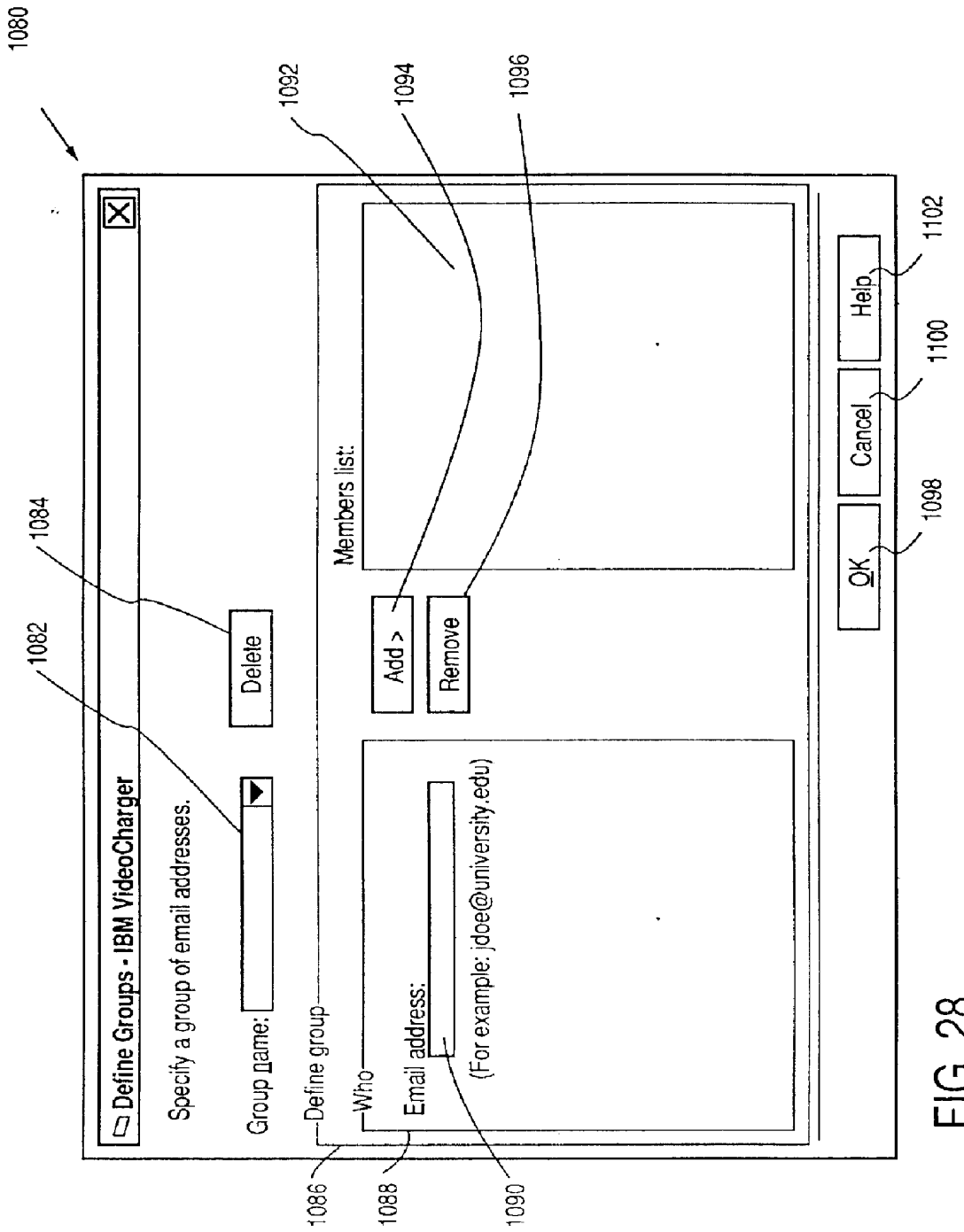

1080
1092
1094
1096
1084
1082
Define Groups - IBM VideoCharger
Specify a group of email addresses.
Group name:
Delete
Define group
Who
Email address:
(For example: jdoe@university.edu)
Add >
Remove
Members list:
OK
Cancel
Help
1102
1100
1098
1086
1088
1090

FIG. 28

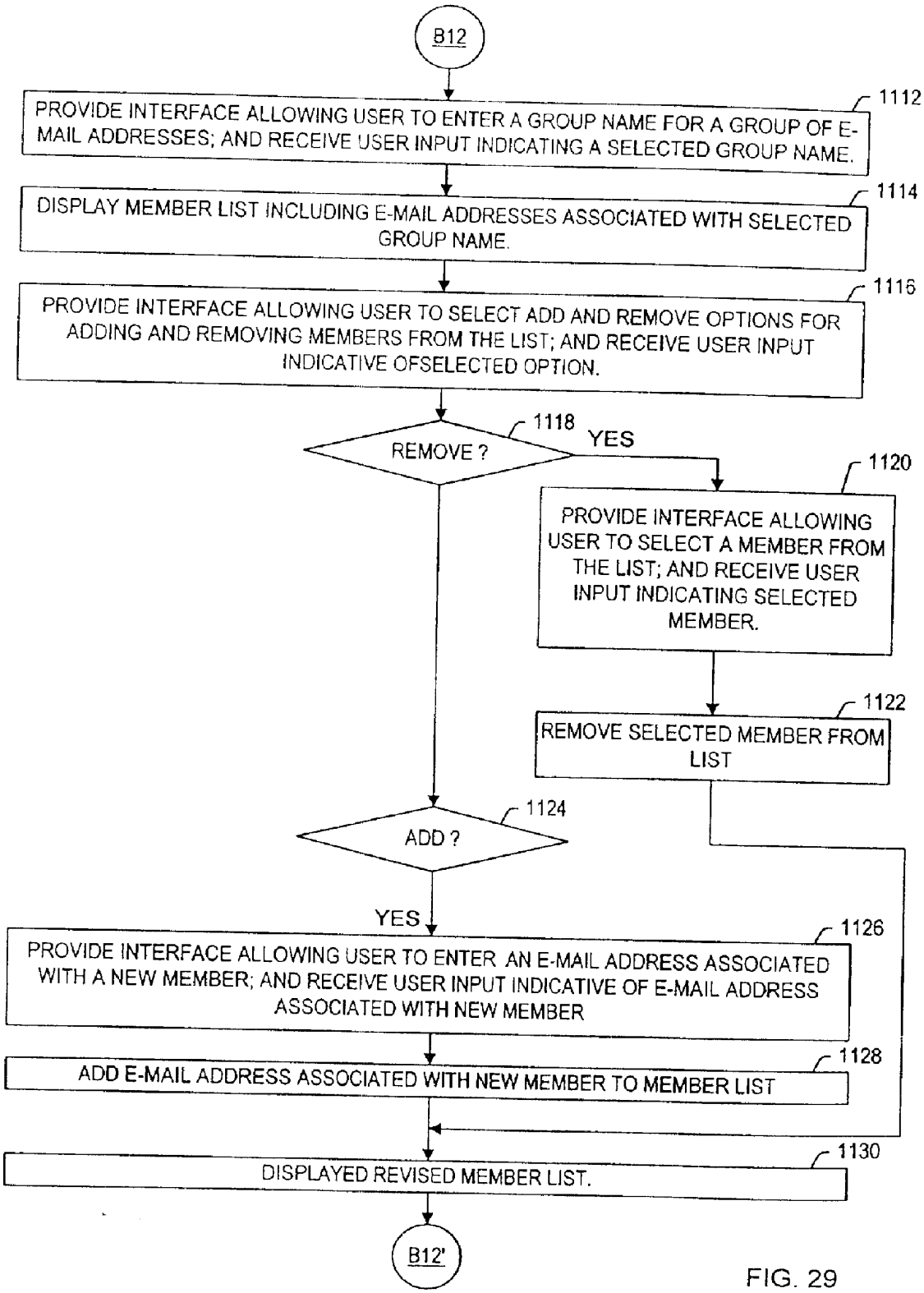

B12
PROVIDE INTERFACE ALLOWING USER TO ENTER A GROUP NAME FOR A GROUP OF E-MAIL ADDRESSES; AND RECEIVE USER INPUT INDICATING A SELECTED GROUP NAME.
1112
DISPLAY MEMBER LIST INCLUDING E-MAIL ADDRESSES ASSOCIATED WITH SELECTED GROUP NAME.
1114
PROVIDE INTERFACE ALLOWING USER TO SELECT ADD AND REMOVE OPTIONS FOR ADDING AND REMOVING MEMBERS FROM THE LIST; AND RECEIVE USER INPUT INDICATIVE OFSELECTED OPTION.
1116
REMOVE ?
1118
YES
PROVIDE INTERFACE ALLOWING USER TO SELECT A MEMBER FROM THE LIST; AND RECEIVE USER INPUT INDICATING SELECTED MEMBER.
1120
REMOVE SELECTED MEMBER FROM LIST
1122
ADD ?
1124
YES
PROVIDE INTERFACE ALLOWING USER TO ENTER AN E-MAIL ADDRESS ASSOCIATED WITH A NEW MEMBER; AND RECEIVE USER INPUT INDICATIVE OF E-MAIL ADDRESS ASSOCIATED WITH NEW MEMBER
1126
ADD E-MAIL ADDRESS ASSOCIATED WITH NEW MEMBER TO MEMBER LIST
1128
DISPLAYED REVISED MEMBER LIST.
1130
B12'

FIG. 29

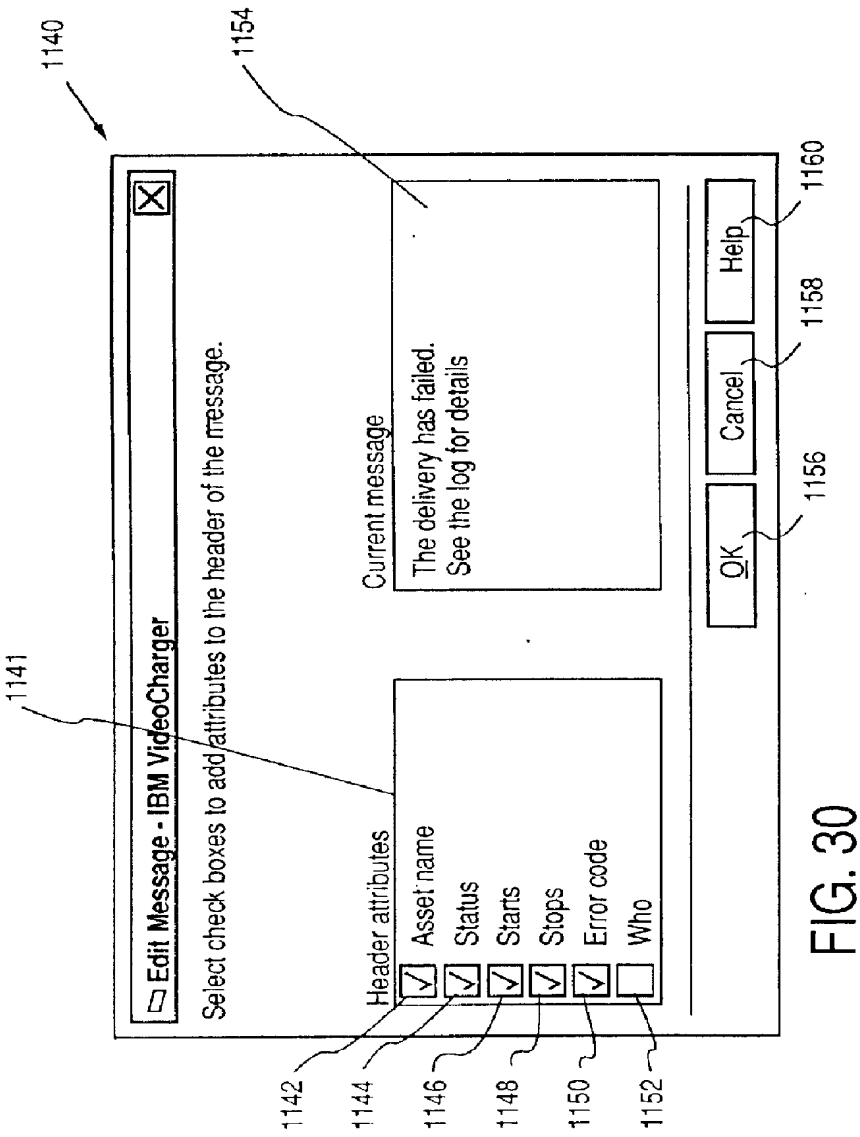

1140
1154
Edit Message - IBM VideoCharger
Select check boxes to add attributes to the header of the message.
Current message
The delivery has failed.
See the log for details
1141
Header attributes
Asset name
Status
Starts
Stops
Error code
Who
1142
1144
1146
1148
1150
1152
OK
Cancel
Help
1156
1158
1160

FIG. 30

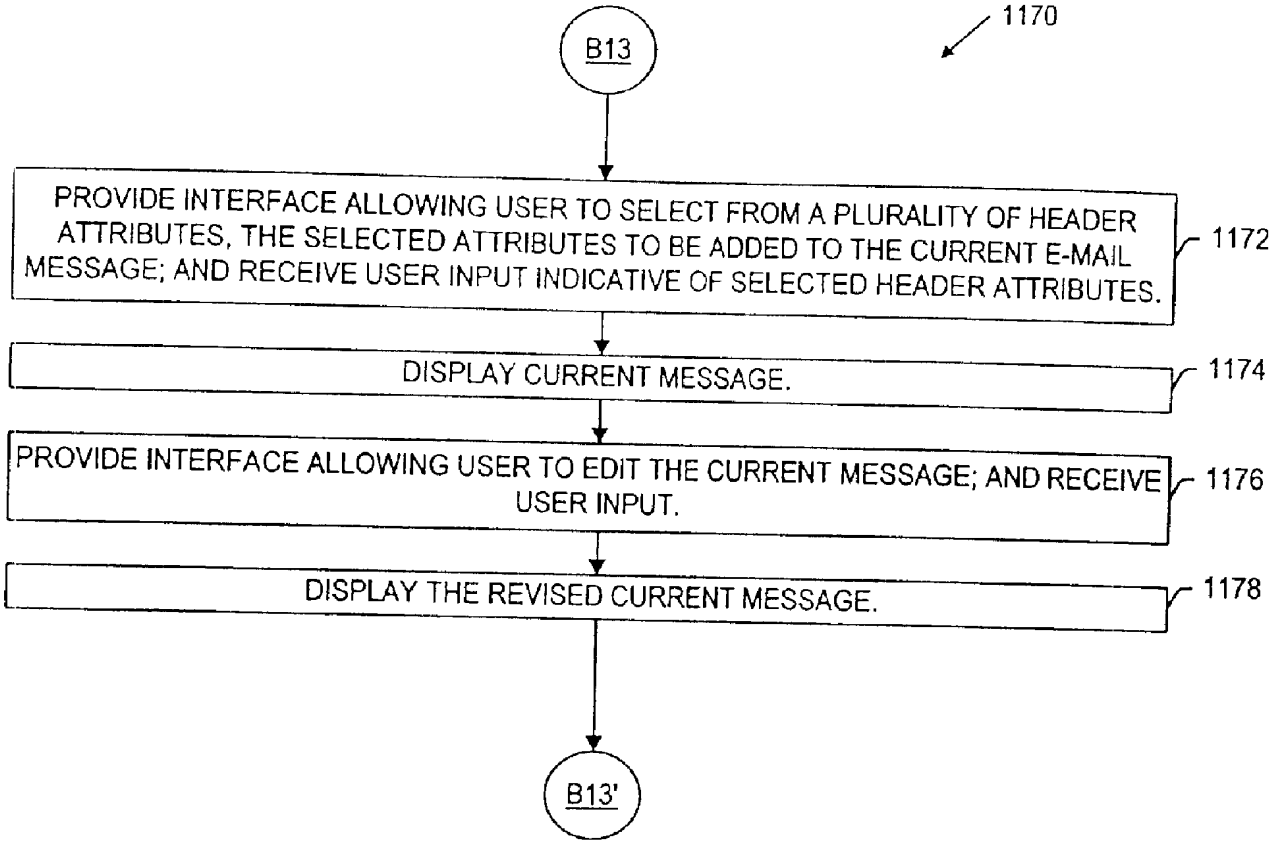

1170
B13
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A PLURALITY OF HEADER ATTRIBUTES, THE SELECTED ATTRIBUTES TO BE ADDED TO THE CURRENT E-MAIL MESSAGE; AND RECEIVE USER INPUT INDICATIVE OF SELECTED HEADER ATTRIBUTES.
1172
DISPLAY CURRENT MESSAGE.
1174
PROVIDE INTERFACE ALLOWING USER TO EDIT THE CURRENT MESSAGE; AND RECEIVE USER INPUT.
1176
DISPLAY THE REVISED CURRENT MESSAGE.
1178
B13'

FIG. 31

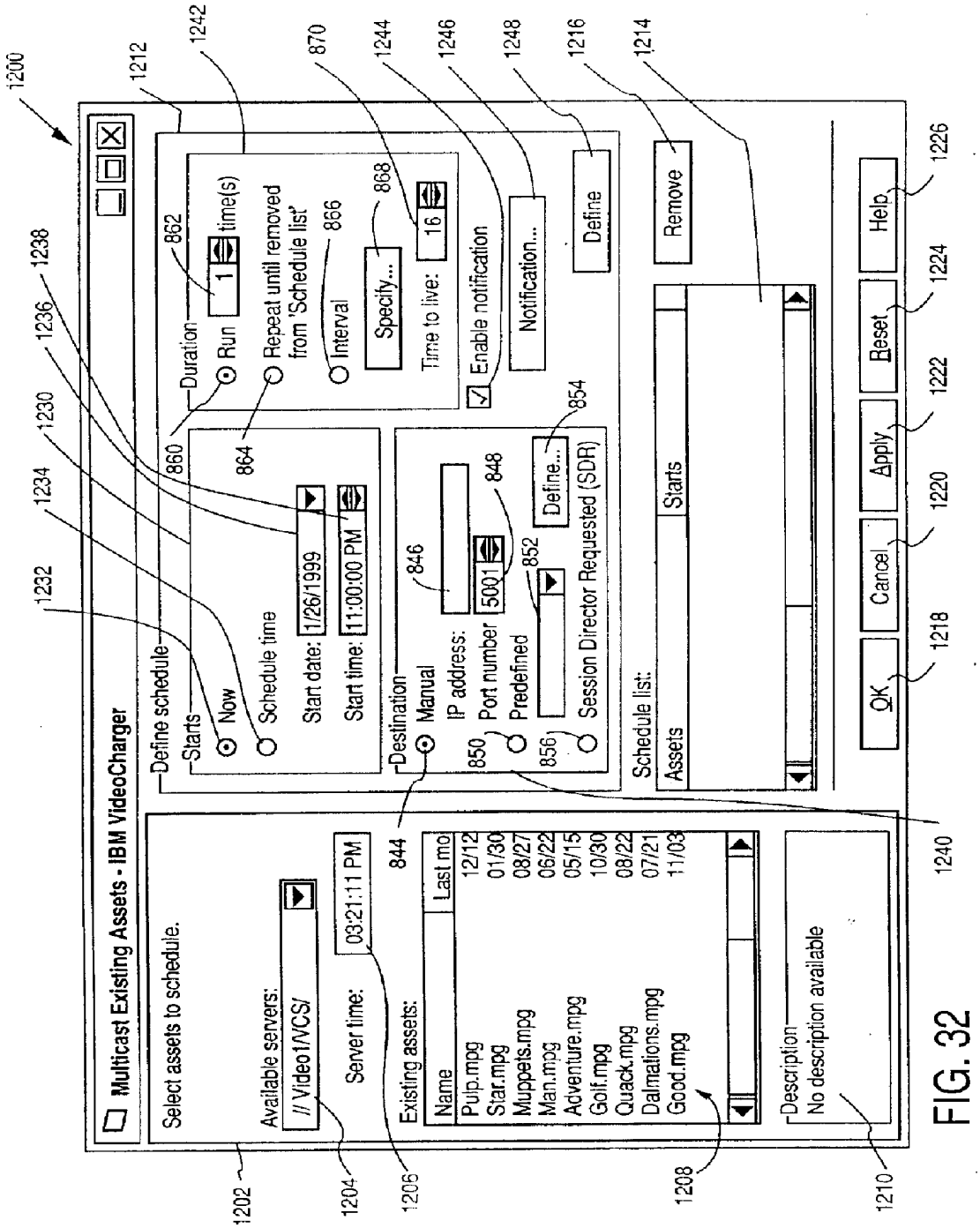

Multicast Existing Assets - IBM VideoCharger
Select assets to schedule.
Available servers:
// Video1/VCS/
Server time:
03:21:11 PM
Existing assets:
Name
Last mo
Pulp.mpg 12/12
Star.mpg 01/30
Muppets.mpg 08/27
Man.mpg 06/22
Adventure.mpg 05/15
Golf.mpg 10/30
Quack.mpg 08/22
Dalmations.mpg 07/21
Good.mpg 11/03
Description
No description available
Define schedule
Starts
Now
Schedule time
Start date: 1/26/1999
Start time: 11:00:00 PM
Duration
Run 1 time(s)
Repeat until removed from 'Schedule list'
Interval
Specify...
Time to live: 16
Enable notification
Notification...
Define
Destination
Manual
IP address:
Port number 5001
Predefined
Define...
Session Director Requested (SDR)
Schedule list:
Assets
Starts
Remove
OK
Cancel
Apply
Reset
Help
1200
1202
1204
1206
1208
1210
1212
1214
1216
1218
1220
1222
1224
1226
1230
1232
1234
1236
1238
1240
1242
1244
1246
1248
844
846
848
850
852
854
856
860
862
864
866
868
870

FIG. 32

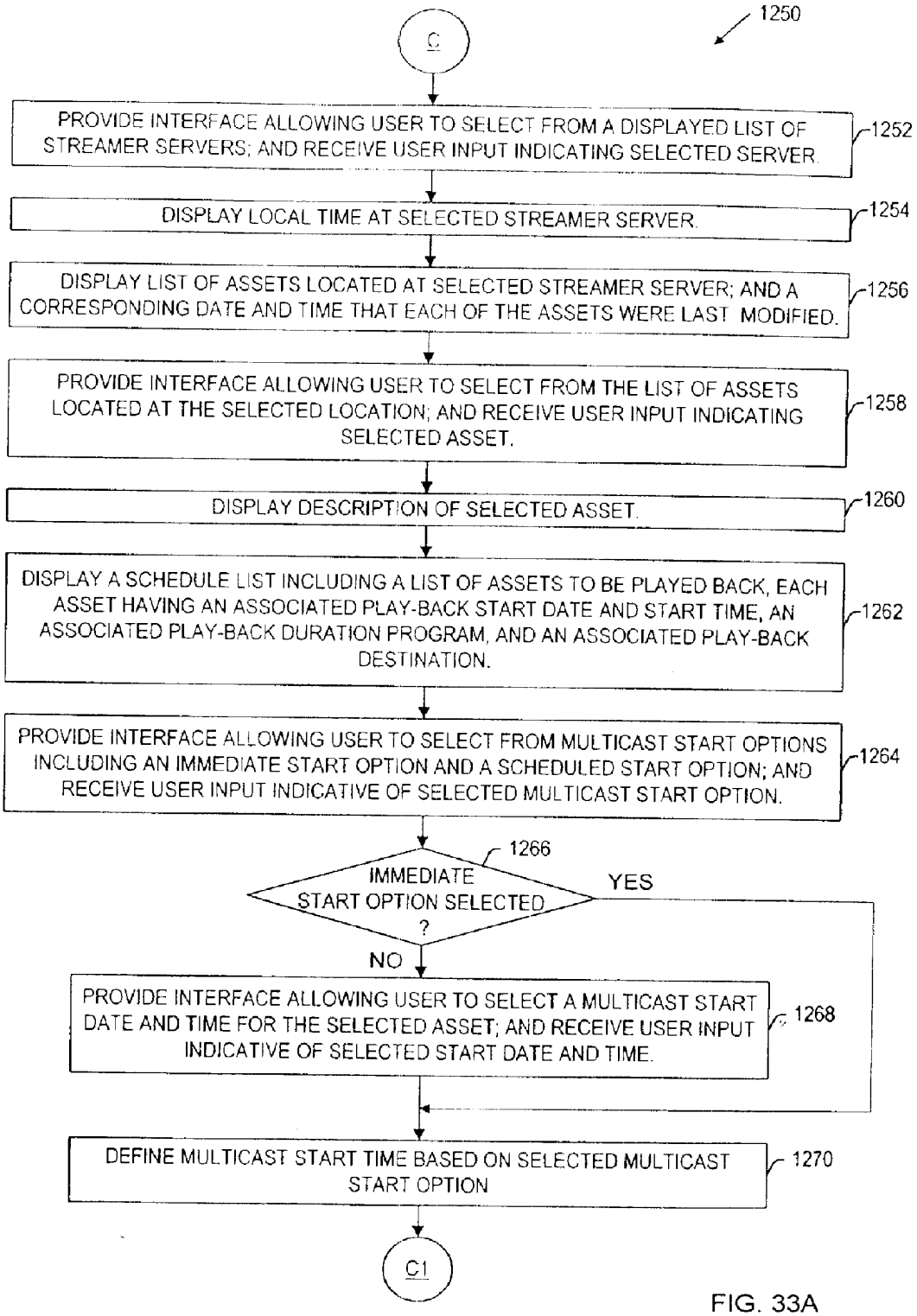

1250
C
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A DISPLAYED LIST OF STREAMER SERVERS; AND RECEIVE USER INPUT INDICATING SELECTED SERVER.
1252
DISPLAY LOCAL TIME AT SELECTED STREAMER SERVER.
1254
DISPLAY LIST OF ASSETS LOCATED AT SELECTED STREAMER SERVER; AND A CORRESPONDING DATE AND TIME THAT EACH OF THE ASSETS WERE LAST MODIFIED.
1256
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM THE LIST OF ASSETS LOCATED AT THE SELECTED LOCATION; AND RECEIVE USER INPUT INDICATING SELECTED ASSET.
1258
DISPLAY DESCRIPTION OF SELECTED ASSET.
1260
DISPLAY A SCHEDULE LIST INCLUDING A LIST OF ASSETS TO BE PLAYED BACK, EACH ASSET HAVING AN ASSOCIATED PLAY-BACK START DATE AND START TIME, AN ASSOCIATED PLAY-BACK DURATION PROGRAM, AND AN ASSOCIATED PLAY-BACK DESTINATION.
1262
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM MULTICAST START OPTIONS INCLUDING AN IMMEDIATE START OPTION AND A SCHEDULED START OPTION; AND RECEIVE USER INPUT INDICATIVE OF SELECTED MULTICAST START OPTION.
1264
1266
IMMEDIATE START OPTION SELECTED ?
YES
NO
PROVIDE INTERFACE ALLOWING USER TO SELECT A MULTICAST START DATE AND TIME FOR THE SELECTED ASSET; AND RECEIVE USER INPUT INDICATIVE OF SELECTED START DATE AND TIME.
1268
DEFINE MULTICAST START TIME BASED ON SELECTED MULTICAST START OPTION
1270
C1

FIG. 33A

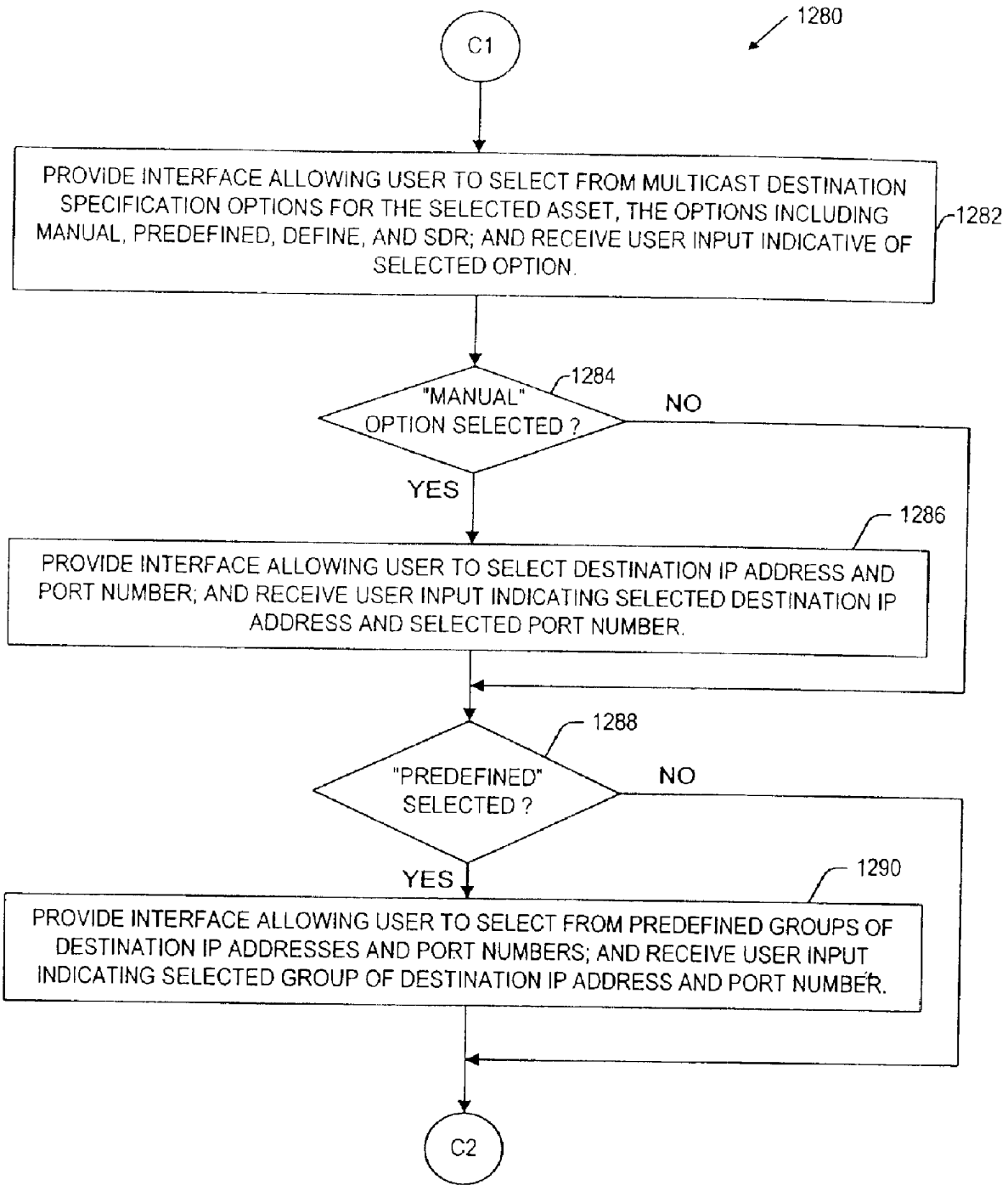

1280
C1
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM MULTICAST DESTINATION SPECIFICATION OPTIONS FOR THE SELECTED ASSET, THE OPTIONS INCLUDING MANUAL, PREDEFINED, DEFINE, AND SDR; AND RECEIVE USER INPUT INDICATIVE OF SELECTED OPTION.
1282
"MANUAL" OPTION SELECTED ?
1284
NO
YES
PROVIDE INTERFACE ALLOWING USER TO SELECT DESTINATION IP ADDRESS AND PORT NUMBER; AND RECEIVE USER INPUT INDICATING SELECTED DESTINATION IP ADDRESS AND SELECTED PORT NUMBER.
1286
"PREDEFINED" SELECTED ?
1288
NO
YES
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM PREDEFINED GROUPS OF DESTINATION IP ADDRESSES AND PORT NUMBERS; AND RECEIVE USER INPUT INDICATING SELECTED GROUP OF DESTINATION IP ADDRESS AND PORT NUMBER.
1290
C2

FIG. 33B

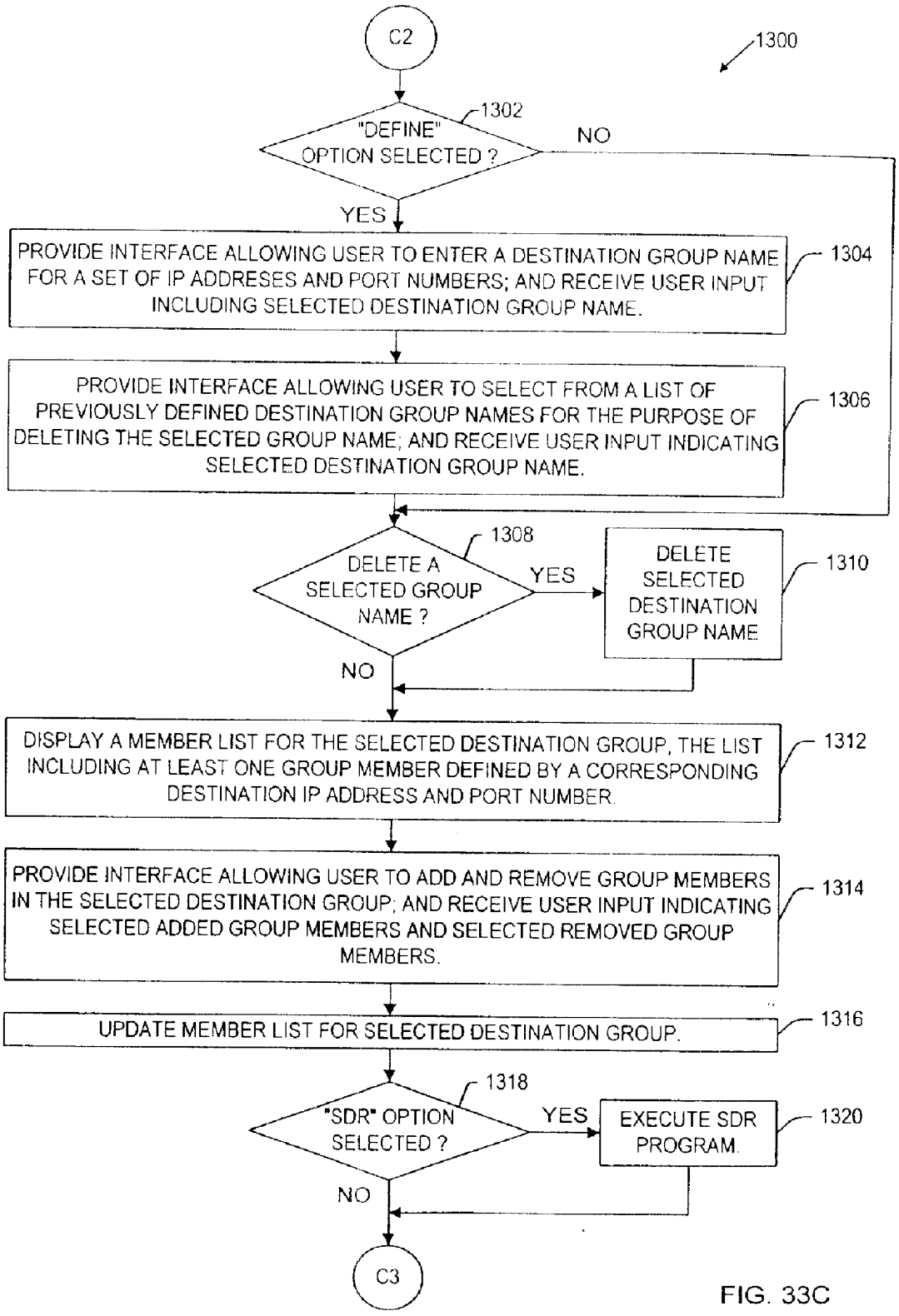

C2
1300
1302
"DEFINE" OPTION SELECTED ?
NO
YES
PROVIDE INTERFACE ALLOWING USER TO ENTER A DESTINATION GROUP NAME FOR A SET OF IP ADDRESES AND PORT NUMBERS; AND RECEIVE USER INPUT INCLUDING SELECTED DESTINATION GROUP NAME.
1304
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A LIST OF PREVIOUSLY DEFINED DESTINATION GROUP NAMES FOR THE PURPOSE OF DELETING THE SELECTED GROUP NAME; AND RECEIVE USER INPUT INDICATING SELECTED DESTINATION GROUP NAME.
1306
1308
DELETE A SELECTED GROUP NAME ?
YES
DELETE SELECTED DESTINATION GROUP NAME
1310
NO
DISPLAY A MEMBER LIST FOR THE SELECTED DESTINATION GROUP, THE LIST INCLUDING AT LEAST ONE GROUP MEMBER DEFINED BY A CORRESPONDING DESTINATION IP ADDRESS AND PORT NUMBER.
1312
PROVIDE INTERFACE ALLOWING USER TO ADD AND REMOVE GROUP MEMBERS IN THE SELECTED DESTINATION GROUP; AND RECEIVE USER INPUT INDICATING SELECTED ADDED GROUP MEMBERS AND SELECTED REMOVED GROUP MEMBERS.
1314
UPDATE MEMBER LIST FOR SELECTED DESTINATION GROUP.
1316
1318
"SDR" OPTION SELECTED ?
YES
EXECUTE SDR PROGRAM.
1320
NO
C3

FIG. 33C

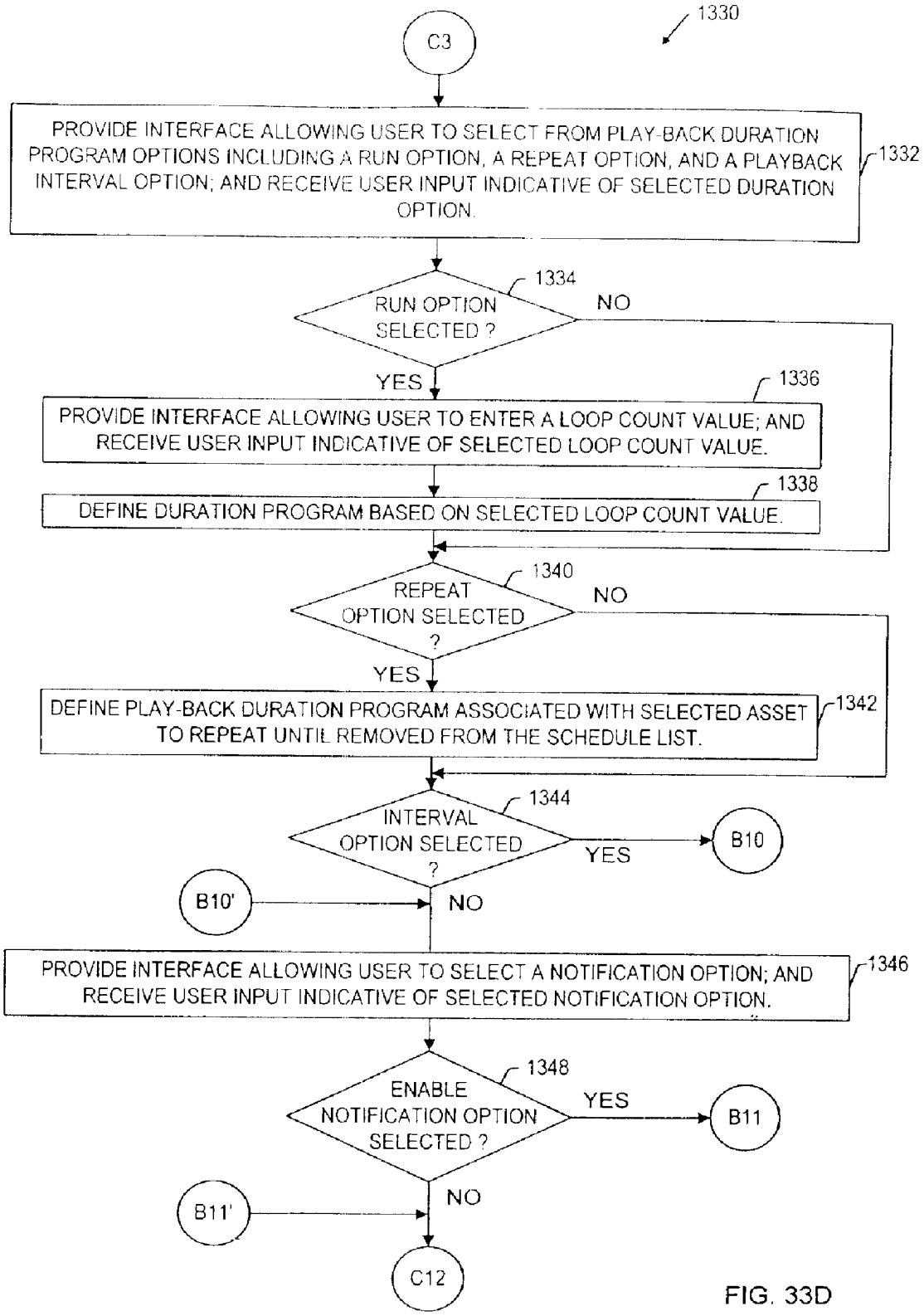

1330
C3
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM PLAY-BACK DURATION PROGRAM OPTIONS INCLUDING A RUN OPTION, A REPEAT OPTION, AND A PLAYBACK INTERVAL OPTION; AND RECEIVE USER INPUT INDICATIVE OF SELECTED DURATION OPTION.
1332
1334
RUN OPTION SELECTED ?
NO
YES
1336
PROVIDE INTERFACE ALLOWING USER TO ENTER A LOOP COUNT VALUE; AND RECEIVE USER INPUT INDICATIVE OF SELECTED LOOP COUNT VALUE.
1338
DEFINE DURATION PROGRAM BASED ON SELECTED LOOP COUNT VALUE.
1340
REPEAT OPTION SELECTED ?
NO
YES
DEFINE PLAY-BACK DURATION PROGRAM ASSOCIATED WITH SELECTED ASSET TO REPEAT UNTIL REMOVED FROM THE SCHEDULE LIST.
1342
1344
INTERVAL OPTION SELECTED ?
YES
B10
B10'
NO
PROVIDE INTERFACE ALLOWING USER TO SELECT A NOTIFICATION OPTION; AND RECEIVE USER INPUT INDICATIVE OF SELECTED NOTIFICATION OPTION.
1346
1348
ENABLE NOTIFICATION OPTION SELECTED ?
YES
B11
B11'
NO
C12

FIG. 33D

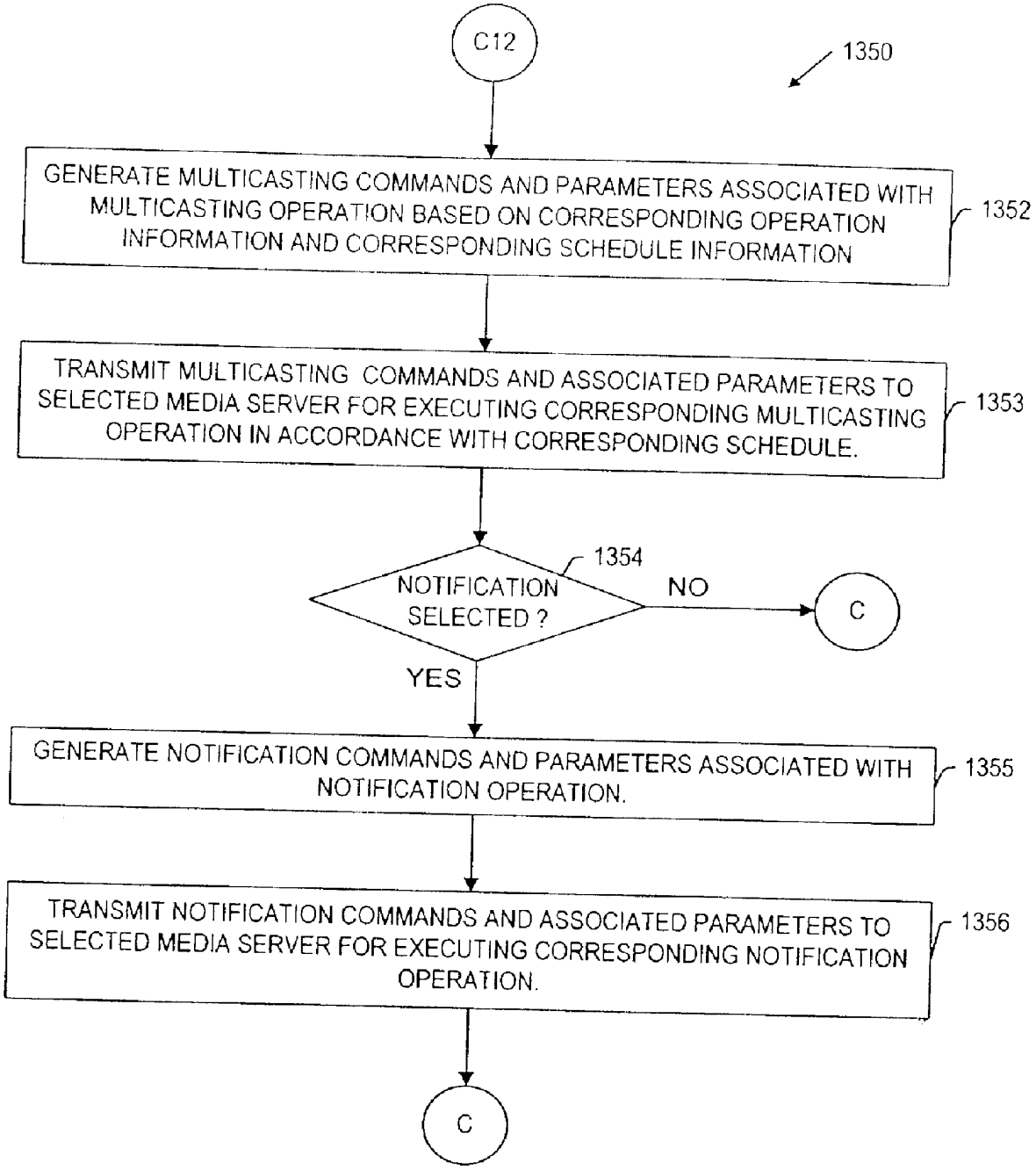

C12
1350
GENERATE MULTICASTING COMMANDS AND PARAMETERS ASSOCIATED WITH MULTICASTING OPERATION BASED ON CORRESPONDING OPERATION INFORMATION AND CORRESPONDING SCHEDULE INFORMATION
1352
TRANSMIT MULTICASTING COMMANDS AND ASSOCIATED PARAMETERS TO SELECTED MEDIA SERVER FOR EXECUTING CORRESPONDING MULTICASTING OPERATION IN ACCORDANCE WITH CORRESPONDING SCHEDULE.
1353
1354
NOTIFICATION SELECTED ?
NO
C
YES
GENERATE NOTIFICATION COMMANDS AND PARAMETERS ASSOCIATED WITH NOTIFICATION OPERATION.
1355
TRANSMIT NOTIFICATION COMMANDS AND ASSOCIATED PARAMETERS TO SELECTED MEDIA SERVER FOR EXECUTING CORRESPONDING NOTIFICATION OPERATION.
1356
C

FIG. 33E

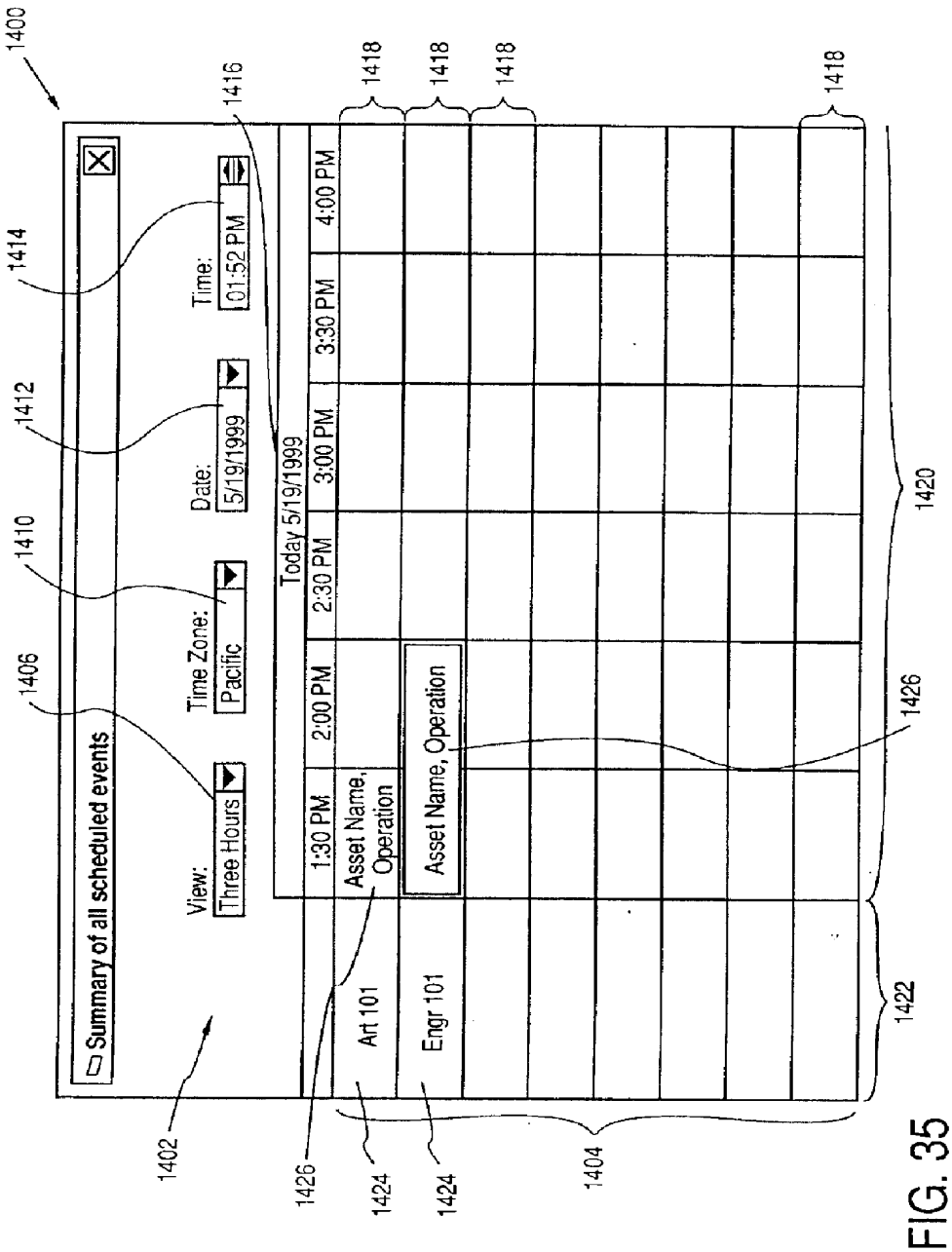

1400
Summary of all scheduled events
View:
Three Hours
Time Zone:
Pacific
Date:
5/19/1999
Time:
01:52 PM
Today 5/19/1999
1:30 PM
2:00 PM
2:30 PM
3:00 PM
3:30 PM
4:00 PM
Asset Name, Operation
Asset Name, Operation
Art 101
Engr 101
1402
1404
1406
1410
1412
1414
1416
1418
1420
1422
1424
1426

FIG. 35

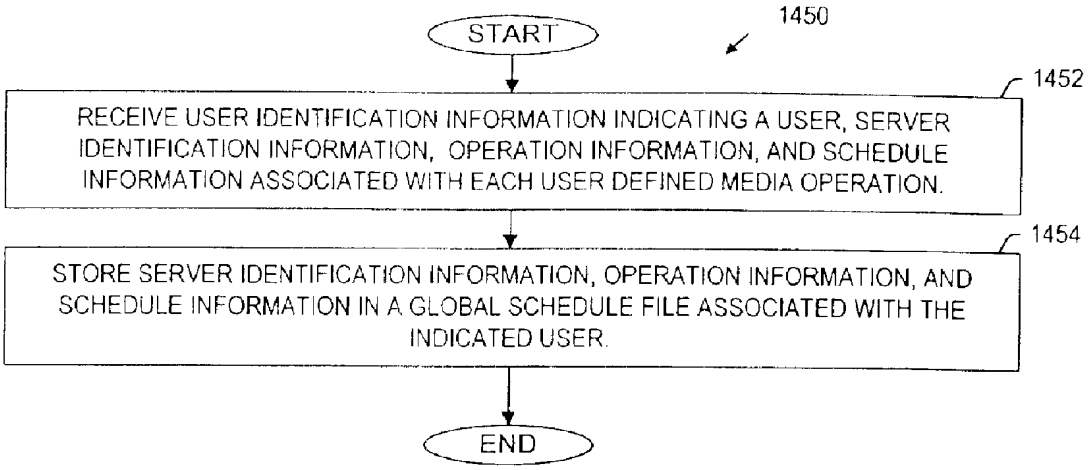

START
1450
1452
RECEIVE USER IDENTIFICATION INFORMATION INDICATING A USER, SERVER IDENTIFICATION INFORMATION, OPERATION INFORMATION, AND SCHEDULE INFORMATION ASSOCIATED WITH EACH USER DEFINED MEDIA OPERATION.
1454
STORE SERVER IDENTIFICATION INFORMATION, OPERATION INFORMATION, AND SCHEDULE INFORMATION IN A GLOBAL SCHEDULE FILE ASSOCIATED WITH THE INDICATED USER.
END

FIG. 36

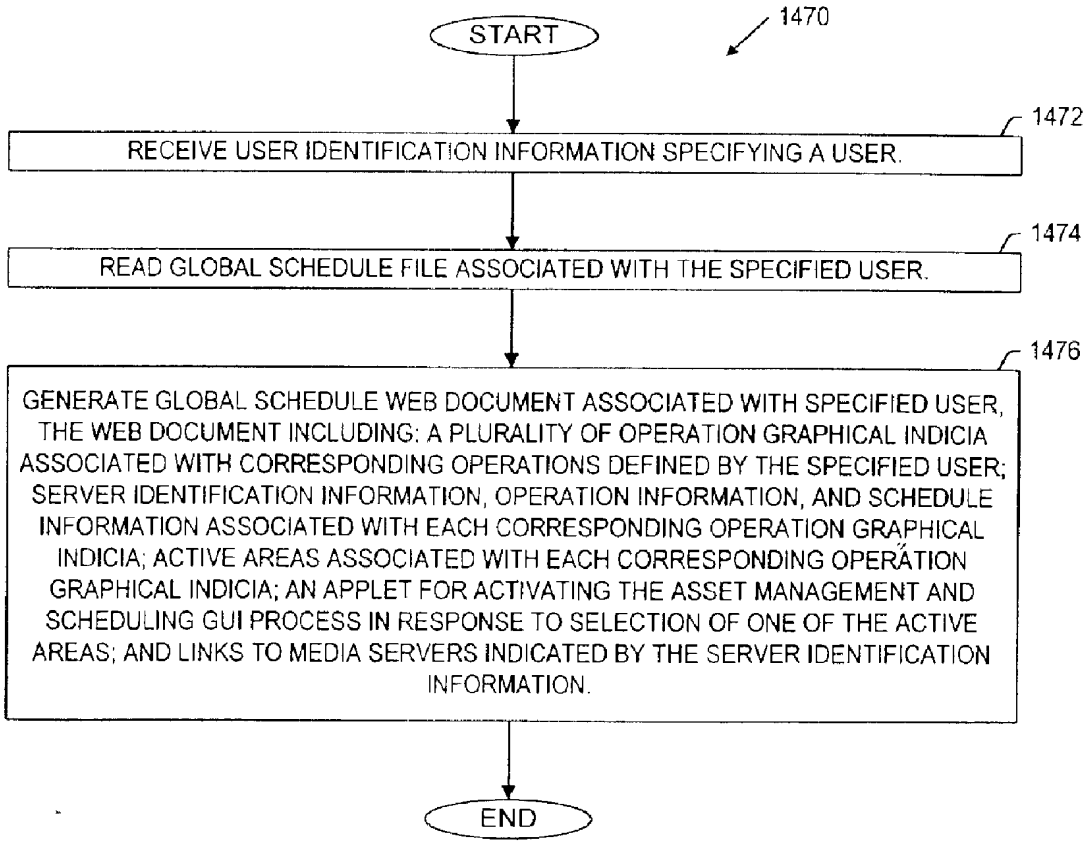

START
1470
1472
RECEIVE USER IDENTIFICATION INFORMATION SPECIFYING A USER.
1474
READ GLOBAL SCHEDULE FILE ASSOCIATED WITH THE SPECIFIED USER.
1476
GENERATE GLOBAL SCHEDULE WEB DOCUMENT ASSOCIATED WITH SPECIFIED USER, THE WEB DOCUMENT INCLUDING: A PLURALITY OF OPERATION GRAPHICAL INDICIA ASSOCIATED WITH CORRESPONDING OPERATIONS DEFINED BY THE SPECIFIED USER; SERVER IDENTIFICATION INFORMATION, OPERATION INFORMATION, AND SCHEDULE INFORMATION ASSOCIATED WITH EACH CORRESPONDING OPERATION GRAPHICAL INDICIA; ACTIVE AREAS ASSOCIATED WITH EACH CORRESPONDING OPERATION GRAPHICAL INDICIA; AN APPLET FOR ACTIVATING THE ASSET MANAGEMENT AND SCHEDULING GUI PROCESS IN RESPONSE TO SELECTION OF ONE OF THE ACTIVE AREAS; AND LINKS TO MEDIA SERVERS INDICATED BY THE SERVER IDENTIFICATION INFORMATION.
END

FIG. 37

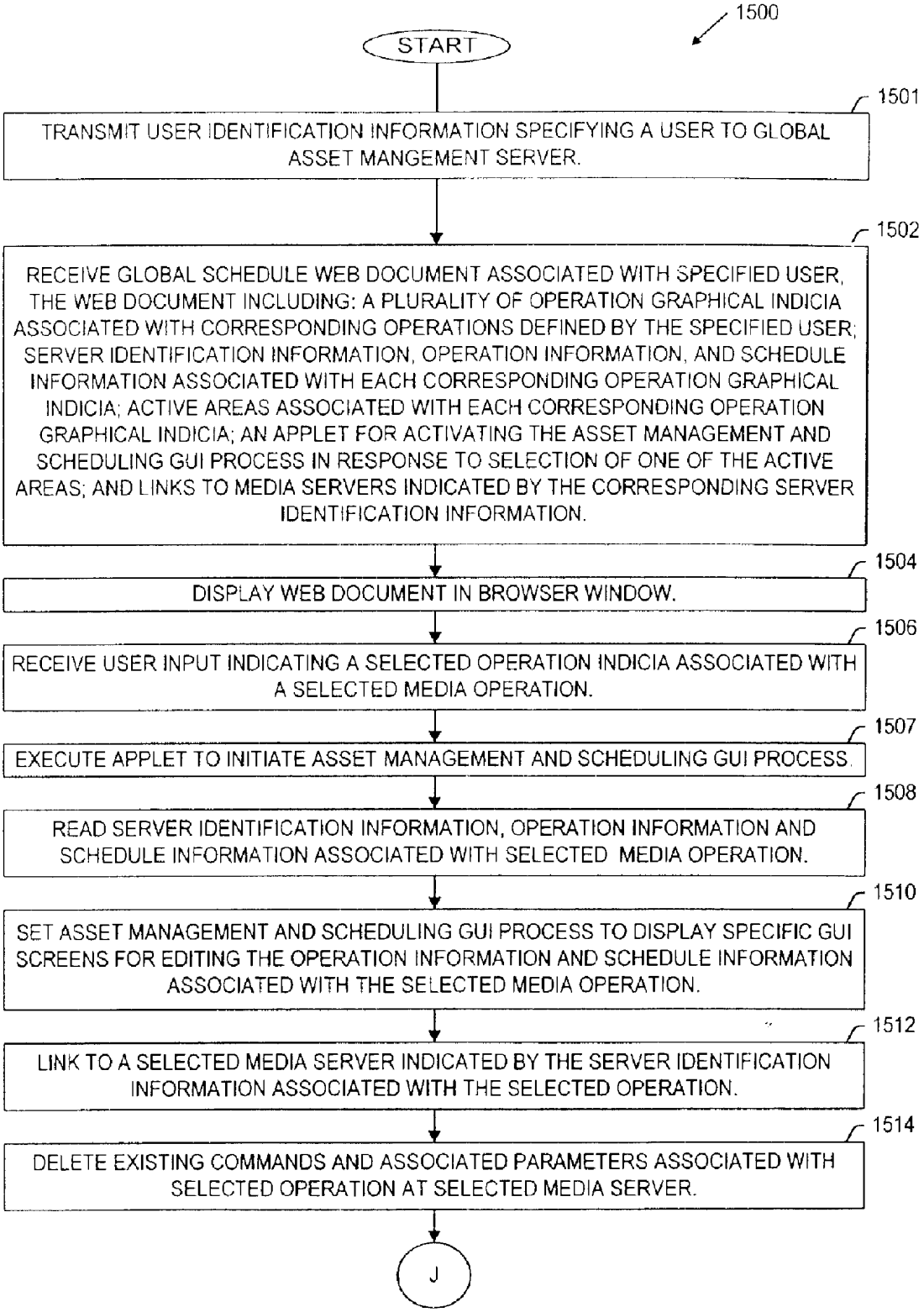

1500
START
1501
TRANSMIT USER IDENTIFICATION INFORMATION SPECIFYING A USER TO GLOBAL ASSET MANGEMENT SERVER.
1502
RECEIVE GLOBAL SCHEDULE WEB DOCUMENT ASSOCIATED WITH SPECIFIED USER, THE WEB DOCUMENT INCLUDING: A PLURALITY OF OPERATION GRAPHICAL INDICIA ASSOCIATED WITH CORRESPONDING OPERATIONS DEFINED BY THE SPECIFIED USER; SERVER IDENTIFICATION INFORMATION, OPERATION INFORMATION, AND SCHEDULE INFORMATION ASSOCIATED WITH EACH CORRESPONDING OPERATION GRAPHICAL INDICIA; ACTIVE AREAS ASSOCIATED WITH EACH CORRESPONDING OPERATION GRAPHICAL INDICIA; AN APPLET FOR ACTIVATING THE ASSET MANAGEMENT AND SCHEDULING GUI PROCESS IN RESPONSE TO SELECTION OF ONE OF THE ACTIVE AREAS; AND LINKS TO MEDIA SERVERS INDICATED BY THE CORRESPONDING SERVER IDENTIFICATION INFORMATION.
1504
DISPLAY WEB DOCUMENT IN BROWSER WINDOW.
1506
RECEIVE USER INPUT INDICATING A SELECTED OPERATION INDICIA ASSOCIATED WITH A SELECTED MEDIA OPERATION.
1507
EXECUTE APPLET TO INITIATE ASSET MANAGEMENT AND SCHEDULING GUI PROCESS.
1508
READ SERVER IDENTIFICATION INFORMATION, OPERATION INFORMATION AND SCHEDULE INFORMATION ASSOCIATED WITH SELECTED MEDIA OPERATION.
1510
SET ASSET MANAGEMENT AND SCHEDULING GUI PROCESS TO DISPLAY SPECIFIC GUI SCREENS FOR EDITING THE OPERATION INFORMATION AND SCHEDULE INFORMATION ASSOCIATED WITH THE SELECTED MEDIA OPERATION.
1512
LINK TO A SELECTED MEDIA SERVER INDICATED BY THE SERVER IDENTIFICATION INFORMATION ASSOCIATED WITH THE SELECTED OPERATION.
1514
DELETE EXISTING COMMANDS AND ASSOCIATED PARAMETERS ASSOCIATED WITH SELECTED OPERATION AT SELECTED MEDIA SERVER.
J

FIG. 38

(19) Canadian Intellectual Property Office
An Agency of Industry Canada

Office de la Propri,t, Intellectuelle du Canada
Un organisme d'Industrie Canada

(11) CA 2 335 662 (13) A1

(40) 03.09.2001
(43) 03.09.2001

(12)

(21) 2 335 662

(51) Int. Cl.7: H04L 29/10, H04L 12/24

(22) 09.02.2001

(30) 09/518,359 US 03.03.2000

(71)
INTERNATIONAL BUSINESS MACHINES CORPORATION,
New Orchard Road, ARMONK, XX (US).

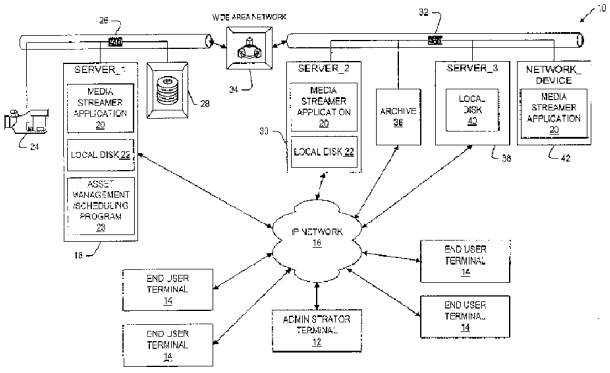
(72)
ROHWER, CARY LANE (US).
BELKNAP, WILLIAM RUSSELL (US).
FARRINGTON, SHANNON MATTHEW (US).
VELIAH, SUNDAR (US).
TAYLOR, LUCY LANE (US).
BENDER, TODD ROBERT (US).

(74)
BARRETT, B.P.

(54) INTERFACE UTILISATEUR GRAPHIQUE SERVANT A LA CREATION DE RESSOURCES
(54) GRAPHICAL USER INTERFACE FOR CREATING ASSETS

(57)

A process is provided for managing operations including encoding operations in a network system including an administrator terminal, at least one server communicatively coupled to the administrator terminal via a computer network, and at least one multimedia device operative to generate media data and being communicatively coupled with a corresponding one of the servers. Each of the servers is operative to selectively activate the corresponding multimedia device, and is also operative to encode media data received from the corresponding multimedia device, and is further operative to access a corresponding memory device for storing media data. The process includes the steps of: displaying a graphical user interface including a server select interface component enabling a user of the administrator terminal to select one of the servers, and a mode select interface component enabling the user to select between a manual encoding mode of operation enabling the user to manually start arid stop a manual encoding process performed by a selected server, and a scheduled encoding mode of operation enabling the user to define a schedule for automatically starting and stopping a scheduled encoding process to be performed by a selected server. User input is received, the input including server identification information indicative of a selected server, mode information indicative of a selected mode of operation, and encoding operation information defined based on the selected mode of operation. Commands and associated parameters, generated based on the encoding operation information, are transmitted to the selected server via the network.



Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2335662 A1 2001/09/03

(21) 2 335 662

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(22) Date de dépôt/Filing Date: 2001/02/09

(41) Mise à la disp. pub./Open to Public Insp.: 2001/09/03

(30) Priorité/Priority: 2000/03/03 (09/518,359) US

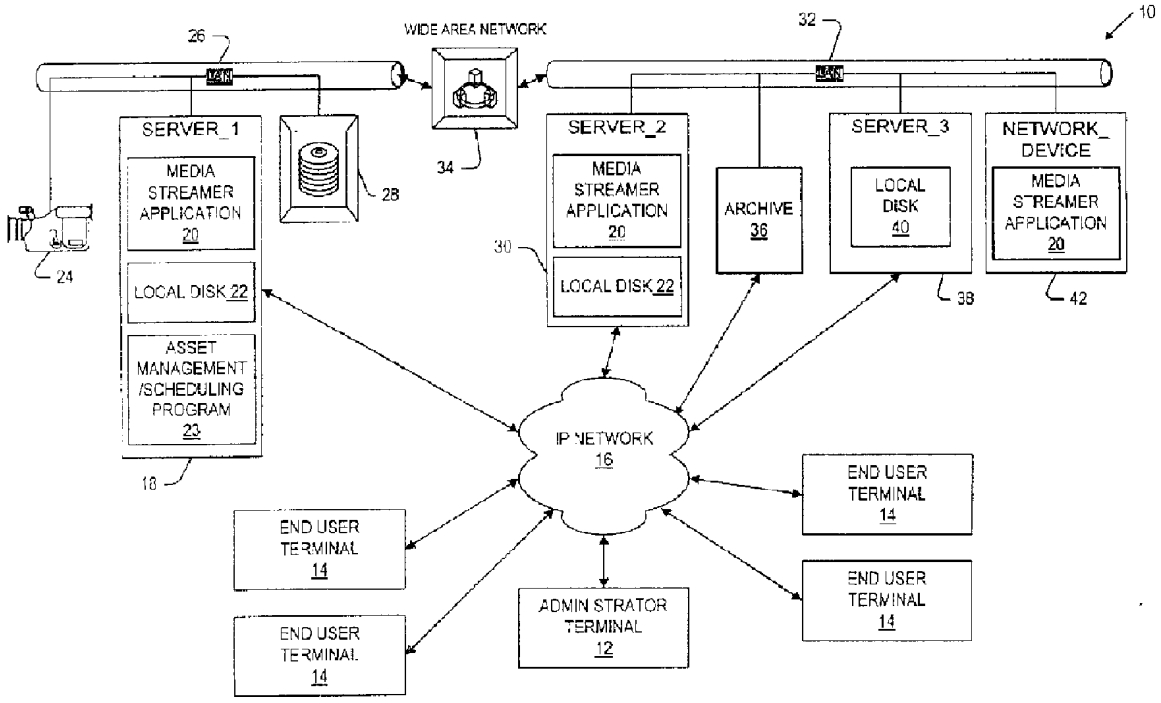
(51) Cl.Int.^{7}/Int.Cl.^{7} H04L 29/10, H04L 12/24

(71) Demandeur/Applicant:
INTERNATIONAL BUSINESS MACHINES CORPORATION, US

(72) Inventeurs/Inventors:
VELIAH, SUNDAR, US;
TAYLOR, LUCY LANE, US;
ROHWER, CARY LANE, US;
FARRINGTON, SHANNON MATTHEW, US;
BENDER, TODD ROBERT, US;
BELKNAP, WILLIAM RUSSELL, US

(74) Agent: BARRETT, B.P.

(54) Titre : INTERFACE UTILISATEUR GRAPHIQUE SERVANT A LA CREATION DE RESSOURCES
(54) Title: GRAPHICAL USER INTERFACE FOR CREATING ASSETS



(57) Abrégé/Abstract:
A process is provided for managing operations including encoding operations in a network system including an administrator terminal, at least one server communicatively coupled to the administrator terminal via a computer network, and at least one multimedia device operative to generate media data and being communicatively coupled with a corresponding one of the servers. Each of the servers is operative to selectively activate the corresponding multimedia device, and is also operative to encode media data received from the corresponding multimedia device, and is further operative to access a corresponding memory device for storing media data. The process includes the steps of: displaying a graphical user interface including a server select interface component enabling a user of the administrator terminal to select one of the servers, and a mode select

(21) 2 335 662

(13) A1

(57) Abrégé(suite)/Abstract(continued):
interface component enabling the user to select between a manual encoding mode of operation enabling the user to manually start arid stop a manual encoding process performed by a selected server, and a scheduled encoding mode of operation enabling the user to define a schedule for automatically starting and stopping a scheduled encoding process to be performed by a selected server. User input is received, the input including server identification information indicative of a selected server, mode information indicative of a selected mode of operation, and encoding operation information defined based on the selected mode of operation. Commands and associated parameters, generated based on the encoding operation information, are transmitted to the selected server via the network.

GRAPHICAL USER INTERFACE FOR CREATING ASSETS

ABSTRACT

A process is provided for managing operations including encoding operations in a network system including an administrator terminal, at least one server communicatively coupled to the administrator terminal via a computer network, and at least one multimedia device operative to generate media data and being communicatively coupled with a corresponding one of the servers. Each of the servers is operative to selectively activate the corresponding multimedia device, and is also operative to encode media data received from the corresponding multimedia device, and is further operative to access a corresponding memory device for storing media data. The process includes the steps of: displaying a graphical user interface including a server select interface component enabling a user of the administrator terminal to select one of the servers, and a mode select interface component enabling the user to select between a manual encoding mode of operation enabling the user to manually start and stop a manual encoding process performed by a selected server, and a scheduled encoding mode of operation enabling the user to define a schedule for automatically starting and stopping a scheduled encoding process to be performed by a selected server. User input is received, the input including server identification information indicative of a selected server, mode information indicative of a selected mode of operation, and encoding operation information defined based on the selected mode of operation. Commands and associated parameters, generated based on the encoding operation information, are transmitted to the selected server via the network.

STL9-1999-0086

GRAPHICAL USER INTERFACE FOR CREATING ASSETS

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates generally to a method and apparatus for managing media data in computer network systems. More specifically, the present invention relates to a process and apparatus providing a centralized graphical user interface for managing media assets in a network by specifying and scheduling operations to be performed by a plurality of media servers in a computer network.

Description of the Prior Art

A media server is a network server system which is operative to perform various operations on media data files in a computer network system. Digital media files may include video data, audio data, text data, and graphical data. Typically, a media server is operative to perform media operations including: copy operations for copying media assets from source locations to destination locations in a network; multicasting operations for streaming media assets from the media server to clients via the network; and encoding operations for encoding media data, received by the media server from multimedia equipment such as a video camera, in order to create media assets.

For purposes of the present application, a media asset is defined to include parsed media data which is configured to be streamed from a media server to one or more clients in a particular type of network. One example of a media server is IBM's VideoCharger™ server and media streamer which combines functions of digital computing and the video broadcast industry into a versatile, cost-effective system for high-quality storage and delivery of multimedia content. Details of the VideoCharger™ product are disclosed in U.S. patent application Ser. No. 08/961,704, entitled "Scaleable Network Transparent Access Framework for Multimedia Serving," filed on or about Oct.

31, 1997, and U.S. patent application Ser. No. 08/961,706, entitled "Seamless Playlist," filed on or about Oct. 31, 1997, each of which is hereby incorporated.

The VideoCharger™ server, provided by International Business Machines Corp., provides one example of a media server. The VideoCharger™ server provides for the delivery of continuous time media data (i.e. audio and/or video data) to Internet or Intranet connected clients. The video is "streamed", (i.e. delivered in real-time) and does not require that the file be downloaded or saved before being played by the client software. In video parlance, the video is "pushed" by the server over the network to the client. This approach is different from most file servers where the data is "pulled" by the client issuing successive "reads" to the server. The "push" architecture is similar to a broadcast environment where a video stream is started by a play command and will continue until stopped. U.S. Patent 5,918,020, issued to Blackard et al. on June 29, 1999 describes a data processing system and a method for implementing a push, or streaming, model for communicating time sensitive encoded data, such as video and audio data, in a communication network. In addition to supporting industry standard file formats for audio and video, the VideoCharger media server supports the popular Internet and World Wide Web (WWW) protocols including IP and Hyper Text Transport Protocol (HTTP). This allows the product to be used with Industry standard applications like HTML Web Browsers. It also allows the product to be used on a wide variety of network types including LANs (Local Area Networks such as Ethernet, Token Ring, FDDI), WANs (Wide Area Networks such as T1, E1, T3, E3) and ATM (Asynchronous Transfer Mode).

An IP multicast feature of the VideoCharger™ media server allows it to be used as a broadcast type server in the Internet environment. This allows a single audio or video stream to be sent to multiple clients, thereby reducing the bandwidth requirements on the network. In addition, VideoCharger on Windows NT offers embedded encoder support. This allows an MPEG encoder to be installed on the server and have VideoCharger directly control the encoder for functions such

as real-time IP Multicast and real-time IP Multicast with live recording of the same stream. This is a very efficient yet powerful method of providing a broadcast of a live event while recording it for later re-broadcast with minimal network load. IBM also provides a VideoCharger Server for AIX.

The functions and capabilities of media servers are very useful to archivists, film/video production groups, educational institutions, research groups, medical technologists, advertising and creative agencies, print and Web publishers, and marketing communications agencies. In accordance with prior art asset management and scheduling methods, an administrator of a media server having access rights thereto may configure the media server to execute specified media operations to be performed by that server in accordance with a specified schedule.

In accordance with one prior art asset management and scheduling method, a administrator may configure a particular media server to perform particular types of media operations by downloading hypertext markup language (HTML) Web pages via corresponding uniform resource locators (URL's) from the particular media server, each of the HTML Web pages providing an interface for scheduling a corresponding one of the particular types of media operations to be executed by the particular media server. A copy/delete HTML Web page provides rudimentary interface functions allowing the administrator to define and schedule copy operations and delete operations to be performed by the particular media server. The copy/delete HTML Web page includes: prompts and active areas enabling the user to specify a copy operation or a delete operation; and text boxes enabling the administrator to enter a path to a selected media file which the subject of the copy or delete operation. An encoding HTML Web page provides rudimentary interface functions allowing the administrator to activate an encoding operation to be performed by the particular media server. The encoding HTML Web page includes text boxes enabling the administrator to enter: a path and a selected file name for the asset to be encoded; and a duration value indicating a duration for the encoding operation. A problem associated with this prior art

method of remotely activating an encoding operation is that the administrator is not provided with any view of the scene to be encoded. A multicasting HTML Web page provides rudimentary interface functions allowing the administrator to schedule a multicasting operation to be performed by the particular media server. The multicasting HTML Web page includes text boxes enabling the administrator to enter: a path indicating a selected asset to be multicasted; one internet protocol (IP) address and port number for multicasting the selected asset; a scheduled start date and start time for executing the multicasting operation; and a duration value indicating a time duration for the multicasting operation.

One problem associated with the above described prior method of configuring media servers to manage media assets and schedule media operations is that there are no mechanisms which prevent the administrator from specifying invalid options, such as specifying an invalid path or an invalid file name for a selected asset. For example, in the course of defining a copy operation using the copy HTML Web page, if the administrator specifies an incorrect path for an asset to be copied, then the copy operation will not be successfully executed at the scheduled time. Also, the prior art interface consisting of the HTML Web pages described above provides no means for notifying the administrator as to whether or not the operation is successfully completed upon passing of the scheduled time for executing the operation. Furthermore, a failure of a particular copy operation would result in a failure of a subsequently scheduled media operation which is dependent upon the particular copy operation, as further explained below.

For a variety of multimedia network applications, it is important to be able to coordinate the scheduling of media operations to be performed by a plurality of media servers. As an example, an administrator of an educational institution such as a college or university may desire to configure one or more media servers to create, store, manage, and stream media assets. In this example, the media assets may include video data and audio data comprising a lecture given by a professor in a

class room. The administrator may configure a first server, that is an encoding server, to encode media data received from multimedia equipment, such as a video camera, at a scheduled time. Alternatively, the administrator may manually control the encoding media server using a local programming interface residing at the server site to start, stop, and pause encoding of the media data generated by the camera while viewing a local screen which displays the lecture being recorded. However, there is no means provided in the prior art for remotely controlling the encoding of media assets by a media server. After the administrator encodes a new asset at an encoding media server, or schedules an encoding operation to be performed at the encoding server, the administrator may configure the encoding server to execute further operations including: storing the encoded asset in a storage device; and multicasting the encoded asset to a selected group of clients either in real time or at a later time in accordance with a programmed schedule. Further, the administrator may configure the encoding server to transfer the encoded asset to a second media server via a network.

In the above described prior art asset management and scheduling method, in order to coordinate the scheduling of media operations at the first and second servers, the administrator must access and configure each server separately. As the number of scheduled operations increases, and the number of configured media servers increases, it becomes more and more difficult for the administrator to coordinate the scheduling of the operations between the multiple servers because there is no centralized location provided in the prior art for automatically logging those operations which are specified and scheduled by the administrator to be executed at the multiple servers. A log of operations scheduled to be executed by each particular media server is stored in a log file maintained by the operating system of the particular media server (e.g., a Windows NT log file, or an AIX log file). However, the administrator must access the log file of each particular server separately in order to review operations to be performed by the particular media server.

In one particular scenario, an administrator may wish to configure a first server to: encode

media data to create a new asset; and then copy the new asset at a later scheduled time to a second server. Provided that the administrator has access to the second server, the administrator may: configure the second server to multicast the encoded asset to a selected group of clients in accordance with a programmed schedule; and configure the second server to subsequently remove the encoded asset from the catalog of the second server so that the new asset cannot be accessed after a certain period of time. The administrator may wish to configure the two servers to perform the steps of creating a new asset at the first media server, transferring the new asset to the second server, and multicasting it from the second server in order to achieve the goal of minimizing network congestion problems which would arise in streaming directly from the first encoding server to the selected clients. Note that the success of the multicasting operation scheduled for execution by the second server is dependent upon the success of the copy operation scheduled for execution by the first server. As mentioned above, a failure of the copy operation would result in a failure of the subsequently scheduled multicasting operation which is dependent upon the copy operation. Because there is no means for notifying the administrator as to whether or not the copy operation is successfully completed, the administrator also would have no warning that the dependent multicasting operation would inevitably fail as well.

In the above described prior art asset management and scheduling method, in order to coordinate the scheduling of media operations at the first and second servers, the administrator could separately access the log files of the first and second servers, and review each. However, this process is very cumbersome. If the administrator needs to coordinate and schedule events at a multiplicity of media servers, which may be residing in different time zones, the coordination and management of the scheduled events at the multiplicity of servers becomes very difficult.

Therefore, one problem with the prior art asset management and scheduling processes is that an administrator of a plurality of media servers is required to interface separately with each media

server in order to read the log files for the purposes of coordinating scheduled operations at each of the media servers. Another problem is that the administrator is prone to specify invalid paths for assets to be operated upon thereby resulting failure of media operations. A further problem is that there is no notification system for notifying the administrator in the event of success or failure of the scheduled operations.

SUMMARY OF THE INVENTION

It is an object of the present invention to provide a graphical user interface process providing a centralized interface for remotely managing media assets, and scheduling media operations to be performed by a plurality of media servers in a computer network system, the media operations including deleting media assets from a source location in a network, copying media assets from selected source locations to selected destination locations; multicasting operations for streaming media assets from selected media servers to selected clients via the network; and encoding operations for encoding media assets.

Briefly, a presently preferred embodiment of the present invention provides a process of managing operations including encoding operations in a network system including an administrator terminal, at least one server communicatively coupled to the administrator terminal via a computer network, and at least one multimedia device operative to generate media data and being communicatively coupled with a corresponding one of the servers. Each of the servers is operative to selectively activate the corresponding multimedia device, and is also operative to encode media data received from the corresponding multimedia device, and is further operative to access a corresponding memory device for storing media data.

The process of managing operations begins with a step of displaying a graphical user interface including a plurality of interface components on a display unit of the administrator

terminal, the components including a server select interface component enabling a user of the administrator terminal to select one of the servers, and a mode select interface component enabling the user to select between a manual encoding mode of operation enabling the user to manually start and stop a manual encoding process performed by a selected server, and a scheduled encoding mode of operation enabling the user to define a schedule for automatically starting and stopping a scheduled encoding process to be performed by a selected server.

The process further includes the steps of: receiving user input via an input/output unit of the administrator terminal, the user input including server identification information indicative of a selected server, mode information indicative of a selected mode of operation, and encoding operation information defined based on the selected mode of operation; generating commands and associated parameters including encoding commands and associated encoding parameters based on the encoding operation information, the encoding commands and associated encoding parameters for instructing the selected server to execute an encoding operation in accordance with the selected mode of operation; and transmitting the commands and the associated parameters to the selected server via the network.

The operations include a preview operation. The administrator terminal also includes a processing unit, and a browser application executed by the processing unit. A selected one of the multimedia devices is a video camera which generates media data including video data. For providing the preview operation, the process further includes the steps of: displaying a preview select interface component enabling the user to select a preview option for executing a preview operation for displaying video data generated by a selected video camera coupled with the selected server; receiving user input indicating that the preview option has been selected; generating preview commands and associated preview parameters for instructing the selected server to execute the preview operation; transmitting the preview commands and the associated preview parameters to the

selected encoding server via the network; receiving preview video data from the selected encoding server via the network; and displaying the preview video data in a browser window on the display unit of the administrator terminal.

If scheduled encoding mode is selected, the steps of displaying a graphical user interface, receiving user input, and generating commands further comprise the steps of: displaying scheduling interface components enabling the user to select a start time and a start date for the scheduled encoding operation; receiving user input indicative of a selected start time and a selected start date for initiating the scheduled encoding operation; and generating encoding commands and associated encoding parameters for instructing the selected encoding server to encode media data received from the selected multimedia device at the selected start time on the selected start date.

Each of the encoding media servers is further operative to record selected portions of media data that are encoded during the encoding operations. The user may select a storage location for storing the encoded media data by browsing a list of available storage locations including predetermined mapped ones of the memory devices associated with the selected encoder media server.

For streaming operations, the network system further includes a plurality of end user terminals communicatively coupled to the administrator terminal and to the encoding servers via the network, each of the encoding servers being further operative to stream media data to selected ones of the end user terminals, the operations further including streaming operations. The process provides an interface for managing streaming operations by performing the steps of: displaying streaming interface components enabling the user to define a streaming operation for streaming the encoded media data to selected ones of the end user terminals; receiving user input including receiving streaming operation information; and generating streaming commands and associated

streaming parameters based on the streaming operation information, the streaming commands and associated streaming parameters for instructing the selected encoding server to execute a streaming operation for streaming the encoded portion of media data from the selected encoder media server to corresponding selected ones of the end user terminals via the network.

The operations which may be managed further include playback operations for streaming the stored portion of encoded media data from the selected encoder media server to corresponding selected ones of the end user terminals in accordance with a user defined playback schedule; and notification operations associated with corresponding ones of the playback operations, the notification operations for sending notification messages to selected network addresses associated with selected ones of the end user terminals and the administrator terminal.

If the manual encoding mode is selected, the process further comprises the steps of: generating manual mode initiation commands and associated parameters for instructing the selected server to initiate a manual encoding operation; transmitting the manual mode initiation commands and associated parameters to the selected server via the network; receiving video data from the selected server via the network; displaying the video data in a browser window on the display unit of the administrator terminal, and also displaying manual encoding mode control interface components enabling the user to select from manual control options including a start option for starting and resuming the manual encoding process, and a stop option for stopping the manual encoding process; and receiving user input indicating a selected one of the manual control options. If the start option is selected, a start command is generated for instructing the selected media server to start the manual encoding process, and the start command is transmitted to the selected server. If the stop option is selected, a stop command for instructing the selected media server to stop the manual encoding process is generated, and the stop command is transmitted to the selected server.

An important advantage of the present invention is that it provides centralized interface provides for managing and scheduling the execution of media operations by a plurality of media servers in different locations.

The foregoing and other objects, features, and advantages of the present invention will be apparent from the following detailed description of the preferred embodiment which makes reference to the several figures of the drawing.

IN THE DRAWINGS

FIG. 1 is a block diagram illustrating a networked computer system including a plurality of network locations for storing and operating on media data, and an administrator terminal communicatively coupled with the locations via a network, the administrator terminal providing an asset management and scheduling graphical user interface (GUI) process in accordance with the present invention for remotely managing operations including generating, copying, and streaming media data at and between selected ones of the locations;

FIG. 2 is a block diagram illustrating an exemplary computer system used to implement the administrator terminal of FIG. 1;

FIG. 3 is a block diagram illustrating a main GUI screen of the asset management and scheduling GUI process, the main GUI screen providing for a user to select from management and scheduling options including managing assets, encoding assets, and multicasting existing assets;

FIG. 4 is a flow diagram illustrating a sub-process for implementing the main GUI screen of FIG. 3;

FIG. 5 is a block diagram illustrating an asset management GUI screen providing for the user

to select from asset management options including mapping new locations, deleting available locations, and copying and deleting selected assets;

FIG. 6 is a flow diagram illustrating a sub-process of the GUI process of the present invention for managing assets using the asset management GUI screen of FIG. 5;

FIGS. 7A through 7E are a block diagrams illustrating location mapping GUI screens providing for the user to map locations including local and remote storage units, media servers, and archives;

FIGS. 8A through 8D are flow diagrams illustrating sub-processes for mapping locations using the GUI screens of FIGS. 7A through 7E;

FIG. 9 is a flow diagram illustrating a sub-process for deleting selected locations from an available locations list;

FIGS. 10A through 10D are flow diagrams illustrating sub-processes for copying and deleting selected assets located at selected locations using the asset management GUI screen of FIG. 5;

FIG. 11 is a block diagram illustrating a rename GUI screen for renaming a selected asset before copying the selected asset;

FIG. 12 is a block diagram illustrating an encoding GUI screen for encoding a new media asset;

FIGS. 13A and 13B are flow diagrams illustrating a sub-process of encoding and optionally recording media data in accordance with the present invention using the GUI screen of FIG. 12;

FIG. 14 is a block diagram illustrating a GUI screen in accordance with the present invention for defining encoder attributes;

FIG. 15 is a flow diagram illustrating a sub-process of defining encoder attributes using the GUI screen of FIG. 14;

FIG. 16 is a block diagram illustrating a GUI screen for manually and remotely controlling the encoding of a new asset;

FIG. 17 is a flow diagram illustrating a sub-process of manually controlling the encoding of an asset from a remote location in accordance with the present invention;

FIG. 18 is a flow diagram illustrating sub-process of defining and scheduling streaming operations in accordance with the present invention;

FIG. 19 is a block diagram illustrating a GUI screen for defining a destination group of IP addresses and port numbers for streaming operations;

FIG. 20 is a flow diagram illustrating a sub-process of defining the destination group of IP addresses and port numbers using the GUI screen of FIG. 19;

FIG. 21 is a flow diagram illustrating a sub-process of specifying and scheduling a recording operation for recording an asset;

FIG. 22 is a block diagram illustrating a playback scheduling GUI screen;

FIGS. 23A through 23D are flow diagrams illustrating a play-back sub-process for defining and scheduling a streaming operation for streaming a new asset using the scheduling GUI screen of FIG. 22;

FIG. 24 is a block diagram illustrating a GUI screen for defining an interval for a playback schedule;

FIG. 25 is a flow diagram illustrating a sub-process for defining a playback schedule interval using the screen of FIG. 24;

FIG. 26 is a block diagram illustrating a GUI screen for defining notification e-mail messages to be sent to selected end users upon detection of a selected event associated with streaming operations;

FIG. 27 is a flow diagram illustrating a process of specifying events for which notification e-mail messages are to be sent to selected end users;

FIG. 28 is a block diagram illustrating a GUI screen for defining groups of destination e-mail addresses for the purpose of sending notification messages to the destination addresses;

FIG. 29 is a flow diagram illustrating a sub-process of defining a group of destination e-mail addresses;

FIG. 30 is a block diagram illustrating an edit message GUI screen;

FIG. 31 is a flow diagram illustrating a process of editing a message using the screen of FIG. 30;

FIG. 32 is a block diagram illustrating a GUI screen providing an interface for defining and scheduling multicasting operations;

FIGS. 33A through 33E are flow diagrams illustrating a sub-process of defining and scheduling streaming operations in accordance with the present invention using the GUI screen of FIG. 32;

FIG. 34 is a block diagram illustrating a GUI screen providing an interface for selecting a date;

FIG. 35 is a block diagram illustrating a global schedule summary Web document in accordance with the present invention;

FIG. 36 is a flow diagram illustrating a process for managing global schedule information in accordance with the present invention;

FIG. 37 is a flow diagram illustrating a process for generating the global schedule Web document of FIG. 35; and

FIG. 38 is a flow diagram illustrating a process for reviewing a summary of all scheduled media operations associated with a particular user of the asset management and scheduling GUI process of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

FIG. 1 shows a block diagram illustrating an exemplary networked computer system at 10 including an administrator terminal 12 configured in accordance with the present invention for implementing an asset management and scheduling GUI process for remotely managing and scheduling media operations including generating, copying, and multicasting media assets at and between a plurality of locations in the system 10 as further explained below.

The depicted system 10 also includes: a plurality of end user terminals 14 communicatively coupled with the administrator terminal 12 via an Internet protocol (IP) network 16 such as the public Internet or a private internet; a first media server 18 designated SERVER_1 communicatively coupled with the administrator terminal 12 via the IP network, the first media server having a media streamer application 20 which is executed by the server, an asset management and scheduling program 23 in accordance with the present invention which is executed by the server, and a local disk 22 providing memory storage for media data including audio data, video data, and text; a video camera 24 for generating media data having video data and audio data, the camera 24 being communicatively coupled with the first media server 18 via a first local area network (LAN) 26; a remote disk 28 communicatively coupled with the first media server 18 via the first LAN 26; a second media server 30 designated SERVER_2 communicatively coupled with the administrator terminal 12 via the IP network, the second server having a media streamer application 20 which is executed by the server, and a local disk 22 for storing media data; an archive data storage unit 36 communicatively coupled with SERVER_2 via a second LAN 32 which is communicatively coupled with the first LAN 26 via a wide area network (WAN) 34; a server 38 designated SERVER_3 communicatively coupled with the administrator terminal 12 via the IP network, and also being coupled with the second LAN 32, the third server having a local disk 40 for storing media data; and a network device 42 having a media streamer application 20 which is executed by the network device 42, the device 42 being communicatively coupled with SERVER_2 via the second LAN 32.

Each of the media streamer applications 20 is operative to encode and parse portions of media data to create a media asset, and is also operative to stream assets to selected groups of the plurality of end user terminals 14. The parsing of the media data includes adding headers and attributes to packets of the media data in order to allow for the detection of failures in staging, that is copying, of media assets from one media server to another media server via the IP network 16 or via the WAN 34. Each of the media streamer applications 20 is executed by a corresponding processing unit (not shown) of the corresponding one of the media servers 18 and 30, and the network device 42.

As further explained below, the end user terminals 14 include plug-in applications which, when executed along with a viewing application such as a browser, are operative to decode and play the assets in real time.

In accordance with the present invention, an asset management and scheduling graphical user interface (GUI) process is provided at the administrator terminal 12 as described in detail below. In general, the asset management and scheduling GUI process allows for a user of the administrator terminal 12 to remotely define and schedule the execution of a variety of operations on media data at selected locations (e.g., selected servers, archives, and networked devices) in the networked computer system 10.

In accordance with the asset management and scheduling GUI process, interface components are displayed on a display unit of the administrator terminal 12, the components prompting a user of the administrator terminal to define and schedule operations to be performed by selected ones of the media servers on corresponding selected assets. In response to the interface components, user input is received via an input/output of the administrator terminal 12. The user input generally includes: operation information specifying details of a selected event, or operation, to be performed

by a selected one of the media servers 18 and 30 on a selected asset which is accessible by the selected media server; and schedule information associated with the operation information, the schedule information indicating an associated schedule for performing the selected operation. In accordance with the asset management and scheduling GUI process, commands and associated parameters are generated based on the operation information and the associated schedule information. The commands and associated parameters are transmitted to the corresponding selected one of the media servers 18 and 30 for execution of the specified operation in accordance with the associated schedule.

Operations which may be remotely managed and scheduled by the asset management and scheduling GUI include: remotely controlling encoding operations for encoding media assets at selected locations in the system 10 by manual remote control, or by scheduling one or more time delayed encoding operations; streaming assets from a selected source media streamer to a selected group of destination end users; "copying" media data from a selected source location in the system 10 setting to a selected destination location; and "deleting" media assets from selected locations.

The media streamer applications 20 are operative to execute several different types of copy actions for copying media assets including: "adding" media files or assets from a local disk 22 to a "catalog" of a corresponding media streamer application 20, the catalog being a list of files and/or assets which the media streamer application may access; "loading" media files or assets from the remote disk 28 to the catalog of the corresponding media streamer application 20 at SERVER_1; and "staging" media assets from a selected source one of the media servers 18 and 30 to a selected destination one of the media servers. In the preferred embodiment, in order to allow for detection of a failure in a staging operation, only media assets which include parsed media data, as opposed to media files which include unparsed media data, are staged via the IP network 16 or via the WAN 34.

The "delete" operations include removing selected assets from the catalog of a selected media streamer application 20 at the selected media server. Media files often comprise intellectual property, and therefore, it is desirable to be able to remove asset from the catalogs of media servers so that the assets cannot be retrieved by persons who are not authorized to do so.

In a scheduled encoding mode, the asset management and scheduling GUI provides for defining and scheduling the activation of the camera 24 in accordance with a user defined schedule to generate video data which is encoded by the media streamer application 20 at SERVER_1. The user may schedule a later date and time for activating the video camera and the encoding process; specify a location to which the encoded data is to be recorded, or stored; specify that the media data is to be streamed out in real time to selected destination locations; and specify that e-mail messages are to be sent to selected end users upon success or failure of the streaming. In a manual encoding mode, the user of the asset management and scheduling GUI controls the encoding process at SERVER_1 while viewing the scene to be encoded from the administrator terminal 12.

In the preferred embodiment of the present invention, the asset management and scheduling GUI process is implemented as a Java applet executed at the administrator terminal 12, the applet being received from the asset management and scheduling program 23 at SERVER_1. In an alternative embodiment of the present invention, the asset management and scheduling GUI process is implemented as a Java application executed at the administrator terminal 12, the application being implemented by executing computer readable instructions stored at the administrator terminal 12. In yet another embodiment of the present invention, the asset management and scheduling GUI process is implemented by logic shared between the administrator terminal 12 and SERVER_1 so as to maximize efficiency.

As further explained below, each of the plurality of end user terminals 14 includes a browser

application running thereon for viewing assets including media data (e.g., an MPEG video clip) streamed to the end user terminal from corresponding ones of the media servers 18 and 30. In one embodiment of the present invention, each of the end user terminals 14 has a media player stored therein, and the browser running on the end user terminal loads the media player in order to view the media data. The media player includes a decoder for decoding the media data. In another embodiment, the browsers running on the end user terminals 14 natively understand the encoding format of the asset streamed thereto and a plug-in is not required.

FIG. 2 shows a block diagram illustrating an exemplary general purpose computer system at 50 which may be used to implement each of the administrator terminal 12, the end user terminals 14, the media servers 18 and 30, and the network device 42 (FIG. 1). The computer system 50 includes: a processing unit 52 communicatively coupled to a bus 54; a random access memory (RAM) unit 56 communicatively coupled to the processing unit via the bus; a read only memory (ROM) unit 58 communicatively coupled to the processing unit via the bus; a main memory unit 60, such as a hard disk or other memory storage device, communicatively coupled to the processing unit via the bus; a display unit 62, such as a cathode ray tube or flat panel display, communicatively coupled to the bus for displaying graphical information; an input/output unit (I/O unit) 64, such as a keyboard and mouse, coupled with the processing unit via the bus, and providing an interface for a user to provide input; and a modem 66 coupled with the bus, and providing communication with remote devices via a computer network such as the IP network (FIG. 1).

In the preferred embodiment of the present invention, the main memory unit 60 (FIG. 2) of the administrator terminal 12 (FIG. 1) stores computer readable instructions for implementing a viewing application supporting a Java interpreter (e.g., Netscape Navigator, Microsoft Explorer, or Sun's HotJava browser). The Java interpreter is operative to execute compiled Java byte code, also called J-code. In the preferred embodiment of the present invention, the asset management and

scheduling GUI process is implemented as a Java applet which is provided to the administrator terminal 12 (FIG. 1) via the IP network 16 (FIG. 1) as executable content inside of a Web document.

Also, in the preferred embodiment of the present invention, the main memory unit 60 (FIG. 2) of each of the end user terminals 14 (FIG. 1) stores computer readable instructions for implementing a viewing application such as a Web browser (e.g., Netscape Navigator, Microsoft Explorer), and computer readable instructions for implementing a plug-in. Further, in the preferred embodiment of the present invention, the main memory unit 60 (FIG. 2) of each of the media servers 18 and 30 (FIG. 1) stores computer readable instructions for implementing the media streamer application 20 (FIG. 1), and instructions for implementing the asset managing and scheduling program 23 (FIG. 1).

With reference back to FIG. 1, initializing steps of the asset management and scheduling process of the present invention include sending a request for a predetermined Web page from the Web browser running on the administrator terminal 12 to the asset management and scheduling program 23 at SERVER_1 via the IP network 16. The management and scheduling program 23 packages the predetermined Web page, along with an asset management and scheduling GUI process applet in accordance with the present invention, for transmission to the Web browser running on the administrator terminal 12. The asset management and scheduling applet is operable to create and manage an embedded menu in a displayed Web page when the Web page is displayed and the applet is executed by the Web browser running on the administrator terminal 12. In the preferred embodiment of the present invention, the asset management and scheduling applet is coded in Java™ and the Web browser supports a Java interpreter.

In a preferred embodiment, each of the media streamer applications 20 provides for encoding media data and parsing the media data to create media assets including encoded and parsed media

data. Also in the preferred embodiment, the media players loaded by the browsers running on the end user terminals 14 are configured to decode and display the media assets streamed thereto by the media streamer application 20.

The asset management and scheduling GUI provided by the administrator terminal 12 is described below with reference to flow diagrams. In the preferred embodiment, the GUI process is implemented as a Java applet executed by the processing unit 52 (FIG. 2) of the administrator terminal 12 (FIG.1) to display a series of GUI screens within a Web browser window on the display unit 62 (FIG. 2) of the administrator terminal. A user of the administrator terminal 12 (FIG. 1) may interface with the below described GUI using the I/O units 64, such as a keyboard and mouse, to provide user input by activating various buttons and check boxes, and by entering and editing text as required and prompted by the GUI screens.

In the below described embodiment of the present invention, each of a plurality of interface functions provided by each of the GUI screens is implemented using a corresponding GUI component selected from a variety of GUI components including radio buttons, check boxes, drop-down lists, spin buttons, editable text boxes, non-editable text boxes, etc. However, as is well understood to one of ordinary skill in the art, each of the interface components of each of the GUI screens described below may alternatively be implemented using other types of GUI components. The asset management and scheduling GUI process of the present invention is more generally described with reference to the flow diagrams while the depicted GUI screens illustrate a currently preferred embodiment.

FIG. 3 shows a block diagram illustrating a main graphical user interface screen (GUI screen) at 100, the screen 100 being used in the asset management and scheduling GUI process of the present invention. The main GUI screen 100 includes an asset management icon 102, an "encode

a new asset" icon 104, and a multicast existing assets icon 106.

FIG. 4 shows a flow diagram illustrating a sub-process at 110 for providing a graphical user interface allowing for a user to select from a plurality of main options of the asset management and scheduling GUI. In the preferred embodiment, sub-process 110 is implemented using the main GUI screen 100 (FIG. 3). The depicted process begins with step 112 in which a user of the administrator terminal 12 (FIG. 1) is prompted to select from main options including an asset management option, an option for encoding a new asset, and an option for multicasting existing assets. In step 114, user input indicating a selected one of the main options is received in response to the user selecting one of the three icons 102, 104, and 106 of the main GUI screen (FIG. 3).

From step 114, the process proceeds to 116 at which it is determined whether the asset management icon 102 (FIG. 3) has been selected, and if so, the process proceeds to "A" (to FIG. 6) to execute an asset management and scheduling GUI sub-process as further explained below. If it is determined at 116 that the asset management option has not been selected, the sub-process proceeds to 118 at which it is determined whether the icon 104 (FIG. 3) for encoding a new asset has been selected, and if so, the depicted process proceeds to "B" (to FIG. 13A) to execute a GUI sub-process for encoding a new asset as further explained below. If is determined at 116 and 118 that neither of the icons 102 and 104 (FIG. 3) of the main GUI screen have been selected, the sub-process proceeds to 120 at which it is determined whether the icon 106 (FIG. 3) for multicasting existing assets has been selected, and if so, the depicted process proceeds to "C" (to FIG. 33A) to execute a GUI sub-process for multicasting existing assets.

FIG. 5 shows a block diagram illustrating an asset management GUI screen at 130, the screen providing an interface for the user to select from asset management options including: mapping, or identifying, new locations (e.g., servers, local and remote disks and archives) to be added to an

available locations list; deleting locations from the available locations list; "copying" selected assets from selected source locations to selected destination locations; and "deleting" selected assets from selected locations as further explained below. As mentioned above, types of locations include local disks, remote disks, media servers, and archives.

The asset management GUI screen 130 includes: an available locations drop-down list 132 for displaying graphical information indicative of the list of available locations from which the user may select a location; a server time display window 134 for displaying the local time at a selected location if the selected location, selected via the drop down list 132, is a server; and an assets list box 136 for displaying graphical information indicative of the names of assets available at the selected one of the available locations indicated in the drop down list 132, the assets in the assets list box 136 being multi-selectable so that the user may select one or more of the listed assets, the assets list box also providing for the display of graphical information indicative of the date and time that each of the listed assets was last modified.

If the selected location, selected via the drop down list 132, is a media server 18, 30 (FIG. 1), or an archive 36 (FIG. 1), then the assets list box 136 displays a list of catalogued assets. "Catalogued" assets include assets in the catalog of a media server, that is assets which are identified by the corresponding media streamer application 20 (FIG. 1) as being accessible. If the scheduled location is a disk (e.g., one of the local disks 22 or the remote disk 28 of FIG. 1), then the assets list box 36 displays all files and assets stored at the selected location. As mentioned above, media files or assets may be added from a local disk 22 (FIG. 1) to a "catalog" of a corresponding media streamer application 20 (FIG. 1), and media files or assets may be loaded from the remote disk 28 (FIG. 1) to the catalog of the corresponding media streamer application 20 (FIG. 1). Upon loading or adding a media file, which includes unparsed media data, the media streamer application 20 (FIG. 1) must parse the media data in order to convert the media file into a media asset which is formatted

for staging and streaming operations.

Note that the server time window 122 is useful to the user because the GUI process is implemented as a Java applet and the user may be residing in a different time zone from the time zone which the selected server resides in. The server time window 122 allows the user to schedule actions for an appropriate time in another time zone.

The screen 130 also includes: a description display box 138 for displaying graphical information indicative of a description of the currently selected asset listed in the assets list box 136; an add/remove locations button 140 for adding locations to the list of available locations and removing locations from the list of available locations as further explained below; a define action box 141 for defining copy actions and delete actions, the box 141 including: a copy assets button 142 for defining a copy action for "copying" a currently selected asset, indicated in the assets list box 136, from a selected source location, identified in the available locations drop down list 132; a delete selected assets button 144 for defining a "delete" action for deleting selected assets at the selected location; and a "copy to" list box 146 allowing for the user to select a destination location from a dynamically created list of possible destination locations, as further explained below.

The asset management and scheduling GUI process includes logic for determining possible destinations for a copy action from a corresponding selected source location. As mentioned above, copy actions for copying media assets include "adding" media files or assets from a local disk 22 (FIG. 1) to a catalog of a corresponding local media streamer application 20 (FIG. 1), "loading" media files or assets from a remote disk 28 (FIG. 1) to the catalog of corresponding media streamer application 20 (FIG. 1), and "staging" media assets from a source media server or a source archive to a destination media server or a destination archive. As mentioned, only media assets, as opposed to media files which include unparsed media data, may be staged. If the selected source location is

a media server or an archive, then it is assumed that the copy action is a staging of the selected asset from the source location a destination location which is to either an archive or another media server. Therefore, if the selected source location is a media server or an archive, then the list of possible destination locations includes only media servers and archives. Alternatively, if the selected source location is a local disk or a remote disk, then it is assumed that the corresponding copy action is an "add" operation or a "load" operation, and the list of possible destination locations includes only media servers because if the scheduled portion of media data is a media file, then the destination media server must parse the media file in order to convert it to an asset and add it, or load it, to the catalog of the destination location.

Note that if a selected destination location is a server or other type of location requiring a log-in name and password, the user is required to enter the log-in name and password.

Note that the delete button 144 is only enabled if the selected location is a media server and the selected portion of media data is an asset. Note also that execution of a "delete" action results in a corresponding selected asset being removed from the catalog of a corresponding media streamer application 20 (FIG. 1) at the corresponding media server.

The GUI screen 130 further includes: a check box 148 for specifying delayed, or scheduled, copying wherein the selected assets are to be copied from the selected source location to the selected destination location at a later time; a start date drop down list, or drop down date picker 150 enabled by checking the check box 148 and providing for the user to select a start date for the scheduled copy action; a start time spin button time picker 152 enabled by checking box 148 and providing for the user to select a start time for the associated schedule copy action; a check box 154 for specifying that the selected asset is to be renamed before copying as further explained below; and a "define" button 155 for adding copy actions and delete actions to the action list box 156 as further explained below.

If box 154 is checked, the GUI process displays a "rename-as" GUI screen (not shown) for the user to enter a target name for the asset upon copying it to the destination location.

The GUI screen 130 further includes an action list box 156 for displaying a list of copy actions and delete actions to be executed as further explained below. The action list box 156 is used for displaying graphical information indicative of a plurality of copy actions and delete actions 160. For each of the actions 160, the action list box 156 also includes: a source name field for displaying graphical information indicative of the corresponding selected asset; a target name 164 for displaying graphical information indicative of the target name, if applicable; a "from" field (not shown) for displaying the selected source location, selected using the drop down list 132; a "to" field (not shown) for displaying the selected destination location, selected via the "copy to" list box 146; and a schedule field (not shown) displaying graphical information indicative of scheduled associated information including the start date and start time selected via components 150 and 152.

The asset management GUI 130 screen further includes: a remove button 166 for removing, or deleting, selected actions from the action list box 156; an OK button 168 for presently causing the execution of, or scheduling the execution of, the actions listed in the actions list 156; a cancel button 170 for canceling execution of the listed actions; an apply button 172 for executing or scheduling execution of the actions listed in the action list 156 without departing from the GUI screen 130; a reset button 174 for returning the GUI process to the state it was in when the user entered the screen or to the state it was in immediately after selecting the apply button (Note that no actions are sent to the server); and a help button 176 for displaying help instructions for using the GUI screen 130. Upon activation of the OK button 168, or the apply button 172, the asset management and scheduling process generates commands and associated parameters specifying the action listed in the action list 156. In the preferred embodiment, the commands and associated parameters are generated by the asset management and scheduling applet executed by the

administrator terminal 12 (FIG. 1) in response to the above described user input. The commands and associated parameters are communicated to the asset management and scheduling program 23 (FIG. 1) at SERVER_1, which then executes the defined copy actions and delete actions for example, the copy actions are executed by copying the selected asset from the source location to the selected destination location on the selected date and time in accordance with the selected renaming option. Delete actions are executed after all copy actions have been executed so that selected assets are not deleted before copying.

FIG. 6 shows a flow diagram illustrating an asset management sub-process at 180. The depicted sub-process proceeds from "A" (from FIG. 4) to step 182 in which an interface is provided, the interface allowing the user to select from asset management options including: mapping, or identifying, new locations in the networked computer system 10 (FIG. 1); deleting available locations listed in the available locations drop down list 132 (FIG. 5); "copying" selected assets from a selected source location to a selected destination location; and "deleting" selected assets from selected locations. In the preferred embodiment, the interface provided in step 182 is implemented using the asset management GUI screen 130 (FIG. 5).

From step 182, the sub-process proceeds to step 184 in which user input indicating a selected one of the asset management options is received. From step 184, the sub-process proceeds to 186 at which it is determined whether the option for mapping a new location has been selected, and if so, the sub-process proceeds to "D" (to FIG. 8A) to execute a location mapping sub-process as further explained below. The location mapping option may be selected by the user by activating the add/remove locations button 140 of the asset management GUI screen 130 (FIG. 5) which causes the asset management process to display an identify location GUI screen as further explained below.

If it is determined at 186 that the identify location option has not been selected, the

sub-process proceeds to 188 at which it is determined whether the delete locations option has been selected, and if so, the process proceeds to "E" (to FIG. 9) to execute a location deleting sub-process as further explained below. Alternatively, if it is determined at 188 that the delete location has not been selected, the sub-process proceeds to 190 at which it is determined whether the option for copying and deleting assets has been selected, and if so the process proceeds to "F" (to FIG. 10A) to execute a sub-process of "copying" and "deleting" selected assets. The user may select the copy option or the delete option by activating the copy button 142 (FIG. 5) or the delete button 144 (FIG. 5) of the asset management GUI screen.

FIG. 7A shows a block diagram illustrating an identify locations GUI screen 200 for mapping new locations for storing assets. The screen 200 is displayed upon activation of the add/remove locations button 140 (FIG. 5) of the asset management GUI screen. The identify locations GUI screen 200 includes: a local disk button 202 for indicating that a local disk (e.g., local disk 22 of SERVER_1 of FIG. 1) is to be mapped; a remote disk button 204 for indicating that a remote disk (e.g., remote disk 28 of FIG. 1) is to be mapped; a media server or archives button 206 for indicating that a media server (E.g., SERVER_1 or SERVER_2 of FIG. 1) or for indicating that an archive (e.g., archive 36 of FIG. 1) is to be mapped; a mapping path window 208 providing for entry and display of graphical information indicating a selected path to a new location; a browse button 210 for browsing predefined paths associated with corresponding ones of the buttons 202, 204, and 206; an available locations display window 212 for displaying available locations; an add button 214 enabled when the path window 208 is not blank, the add button 214 for adding a location, indicated by the path displayed in the path window 208, to the available locations list and window 212; and a remove button 216 for specifying that a location is to be deleted from the available locations window 212 and the drop down list 132 (FIG. 5).

The identify locations GUI screen 200 also includes an OK button 218 for executing the

selected add and remove actions defined using the above described GUI components of screen 200, an apply button 222 for executing the add and remove actions without departing from the screen 200, and a reset button 224, each of which is enabled when the add button 214 is selected. Upon activation of the OK button 218, or the reset button 224, the newly mapped locations are added to the available location drop down list 132 (FIG. 5).

FIG. 7B shows a block diagram illustrating file transfer protocol (FTP) information interface components 227 of the identify locations GUI screen 200, the components 227 being displayed and enabled upon activation of the remote disk button 204 which indicates that a remote disk is to be mapped. The FTP information interface components 227 include: a user ID text entry box 228 providing for the user to enter a user ID for accessing a remote disk indicated by the path entered in the path window 208; and a password entry box 228 providing for the user to enter a password for accessing the remote disk.

FIG. 7C shows a block diagram illustrating a "browse local disks" GUI screen 230 for mapping a path to a local disk and directory, such as for one of the local disks 22 (FIG. 1) of the media servers 18 and 30 (FIG. 1). The "browse local disk" GUI screen 230 is activated and displayed in response to the user activating the browse button 210 (FIG. 7A) of the identify location screen while the local disk button 202 (FIG. 7A) is activated. The "browse local disk" GUI screen 230 includes: a display window 232 for displaying graphical information indicative of folders to be mapped in a directory of a local disk, the folders being selectable by the user; an OK button 234; a cancel button 236; and a help button 238. The user may select a folder from the folders displayed in window 232, the selected folder being identified as a mapped location. Activating the OK button 234 after one of the folders has been selected causes the GUI process to map the selected folder and then return to the identify locations GUI screen 200 (FIG. 7A), and a path to the selected folder is displayed in the path window 208 (FIG. 7A). The user may then add the selected folder to the

available location list box 212 (FIG. 7A) of the identify location GUI screen by activating the add button 214 (FIG. 7A). Activating the cancel button 236 cancels the users selections, if any, and returns the GUI process to the identify location GUI screen 200 (FIG. 7A) without any action. Activating the help button 238 provides graphical information for assisting the user in operating the GUI process.

FIG. 7D shows a block diagram illustrating a "browse remote drives" GUI screen 240 for mapping a path to a remote disk and directory (e.g., remote disk 28 of) FIG. 1. The "browse remote drives" GUI screen 240 is activated and displayed in response to the user activating the browse button 210 (FIG. 7A) of the identify locations screen while the remote disk button 204 (FIG. 7A) is activated. The "browse remote drives" GUI screen 240 includes: a display window 242 for displaying graphical information indicative of folders in a directory of a remote disk, the folders being selectable by the user; an OK button 234; a cancel button 236; and a help button 238. The user may select a folder from the folders displayed in window 242, the selected folder being identified as a mapped location. Activating the OK button 244 returns the GUI process to the identify locations GUI screen 200 (FIG. 7A), and a path to the selected folder is displayed in the path window 208 (FIG. 7A). The user may then add the selected folder to the available location list box 212 (FIG. 7A) by activating the add button 214 (FIG. 7A). Activating the cancel button 246 cancels the users selections, if any, and returns the GUI process to the identify location GUI screen 200 (FIG. 7A) without any action.

FIG. 8A shows a flow diagram illustrating an identify location sub-process 250 in accordance with the present invention. The depicted sub-process proceeds from "D" (from FIG. 6) to step 252 in which an interface is provided for allowing the user to select from mapping options including: mapping a path to a local disk such as one of the local disks 22 (FIG. 1) of the media servers 18 and 30 (FIG. 1); mapping a path to a remote disk such as the remote disk 28 (FIG. 1) associated with

SERVER_1 (FIG. 1); and a mapping a media server (e.g., SERVER_1 or SERVER_2 of FIG. 1) or archive (e.g., archive 36 of FIG. 1). From step 252, the sub-process proceeds to step 254 in which user input indicative of a selected one of the mapping options is received. In the preferred embodiment, the GUI screen 200 (FIG. 7A) prompts the user to select one of the buttons 202, 204, and 206 (FIG. 7A) in steps 252 and 254.

From step 254, the sub-process proceeds to 256 at which it is determined whether the local mapping option has been selected, that is whether button 202 (FIG. 7A) has been selected, and if so, the process proceeds to "D1" (to FIG. 8B). If it is determined at 256 that the local mapping option has not been selected, the process proceeds to 258 at which it is determined whether the remote mapping option has been selected, that is whether button 204 (FIG. 7A) has been selected. If it is determined at 258 that the remote mapping option has been selected, the process proceeds from 258 to "D2" (to FIG. 8C). If it is determined at 256 and 258 that neither the local mapping option nor the remote mapping option has been selected, the sub-process proceeds to 260. At 260, it is determined whether the "media server or archive mapping option" has been selected, that is whether button 206 (FIG. 7A) has been activated, and if so, the process proceeds to "D3" (to FIG. 8D). Alternatively if it is determined at 260 that the media server or archive mapping option has not been selected, the process proceeds to "D'" (back to FIG. 6).

FIG. 8B shows a flow diagram illustrating a sub-process at 270 for mapping a path to a local disk for storing media assets. The sub-process 270 proceeds from "D1" (from FIG. 8A) to step 272 in which the user is prompted to define a path to a new local disk, and directory if applicable. From step 272, the sub-process proceeds to 274 in which information indicative of a path to the new local disk and directory is received. In the preferred embodiment, steps 272 and 274 are implemented using the identify locations GUI screen 200 (FIG. 7A) wherein the user provides user input indicating a path to a new local disk by entering text indicative of a path in the path window 208

(FIG. 7A). Also in the preferred embodiment, the user may optionally activate the browse button 210 (FIG. 7A) which causes the GUI process to display the browse local disks screen 230 (FIG. 7C) which assists the user in indicating a path to a new local disk as explained above. From step 274, the sub-process proceeds to step 276 in which a new available location defined by the path is added to an available location list. From step 276, the sub-process proceeds to step 278 in which a revised available locations list is displayed. In the preferred embodiment, the revised available locations list is displayed in the available locations window 212 (FIG. 7A).

FIG. 8C shows a flow diagram illustrating a sub-process at 280 for mapping and identifying a new remote disk as one of the available locations. The depicted sub-process proceeds from "D2" (from FIG. 8A) and proceeds to step 282 in which the user is prompted to enter: a path to a new remote disk and directory; and a user ID and password for accessing the new remote disk if required. From step 282, the sub-process proceeds to step 284 in which user input indicative of a path, a user ID, and a password is received. From step 284, the sub-process proceeds to step 286 in which a new remote location is added to the available location list. In step 288, a revised location list is displayed. From step 288, the process proceeds back to step "D'" (back to FIG. 8A). In the preferred embodiment, the sub-process 280 is implemented using the identify locations GUI screen 200 (FIG. 7B).

FIG. 8D shows a flow diagram illustrating a sub-process at 290 for defining a path to a media server, such as one of the media servers 18 and 30 (FIG. 1), or an archive such as the archive 36 (FIG. 1). The sub-process 290 proceeds from "D3" (from FIG. 8A) to step 292 in which the user is prompted to enter a path to a new media server or archive location. From step 292, the sub-process proceeds to step 294 in which user input indicative of a path to a new media server or archive location is received. In the preferred embodiment, steps 292 and 294 are implemented using the identify locations GUI screen 200 (FIG. 7A) wherein the user provides user input indicating a

path to a new media server or archive location by entering information in the parameter path window 208 (FIG. 7A). From step 294, the sub-process proceeds to step 296 in which the path to the new remote location specified in step 294 is added to the available locations list displayed in the available locations window 212 (FIG. 7A) by activating the add button 214 (FIG. 7A). In step 298, the revised available location list is displayed in window 212 (FIG. 7A). From 298, the sub-process proceeds to "D3'" (back to FIG. 8A). Note that a user of the asset management and scheduling GUI process at the administrator terminal must be authorized to access locations such as servers. Particular locations (e.g., SERVER_1, archive 36, and SERVER_2) may require a log-in name and password in order to access the location. The user is required to enter a log-in name and password upon commiting/defining an action to a server.

FIG. 9 shows a flow diagram illustrating a sub-process at 302 for deleting selected locations from the available locations list displayed in window 212 (FIG. 7A) of the identify locations GUI screen 200. The sub-process 302 proceeds from "E" (from FIG. 6) to step 302 in which a list of available locations is displayed. In the preferred embodiment, the available locations list is displayed in window 212 (FIG. 7A) of the identified location GUI screen. From step 302, the sub-process proceeds to step 304 in which an interface is provided allowing the user to select locations to be removed from the available locations list. From step 304, the sub-process proceeds to step 306 in which user input indicating a selected one of the locations to be removed is received. From step 306, the sub-process proceeds to step 308 in which the selected location, or locations, are removed from the available locations list. In step 310, a revised available locations list is displayed. In the preferred embodiment, the sub-process 302 is implemented using the identify locations GUI screen 200 (FIG. 7A) wherein the user selects one or more locations from the locations displayed in the window 212 (FIG. 7A) and activates the remove button 216 which causes the selected locations to be removed from the list, and window 212 (FIG. 7A) then displays the revised available locations list. From step 310, the process proceeds back to "E'" (back to FIG. 6).

FIG. 10A shows a flow diagram illustrating a sub-process at 320 for "copying" and "deleting" selected assets. The sub-process 320 proceeds from "F" (from FIG. 6) to step 322 in which an interface allowing the user to select from a displayed list of available locations is provided. The displayed list includes: locations of local disks such as local disk 22 (FIG. 1); locations of remote disks such as remote disk 28 (FIG. 1); locations of media servers such as SERVER _1 or SERVER _2 (FIG. 1); and locations of archives such as archive 36 (FIG. 1). Also in step 322, user input indicating a selected one of the locations is received. In the preferred embodiment, the sub-process 320 of is implemented using the asset management GUI screen 130 (FIG. 5). Therefore, in the preferred embodiment, step 322 is implemented using the available locations drop down list 132 (FIG. 5) which provides the list of available locations from which the user may select as described above.

Note that upon selecting a location, if the user has not already logged on to the server associated with the selected location, the asset management and scheduling process executes instructions at the user terminal 12 (FIG. 1) for accessing the selected server associated with the selected location (e.g., SERVER_1 or SERVER_2 of FIG. 1). Note that the user may be required to enter a log-in name and password in order to establish a connection with the selected server.

From step 322, the sub-process proceeds to 324 at which it is determined whether the location selected in step 322 is a server, and if so, the sub-process proceeds to step 326 in which the local time at the selected server is displayed. In the preferred embodiment, the local time of the selected server is displayed in the server time window 134 (FIG. 5) of the asset management GUI screen. As mentioned above, the server time window 122 is useful to the user because the GUI process is implemented as a Java applet and the user may be residing in a different time zone from the time zone which the selected server resides in. The server time window 122 allows the user to schedule actions for an appropriate time in another time zone.

From step 326, the sub-process proceeds to step 328 in which a list of assets located at the selected location is displayed along with a corresponding date and time that each of the displayed assets were last modified. In the preferred embodiment, these assets are displayed in the assets window 136 (FIG. 5) of the asset management GUI screen. In step 330, an interface allowing the user to select from the list of assets located at the selected location is provided, and user input indicating a selected one of the assets is received. In the preferred embodiment, the user may select one of the assets in the assets list box 136 (FIG. 5) by clicking on it. Note that the assets list box 136 (FIG. 5) is multi-selectable.

In step 332, a description of the selected one of the assets located at the selected location is displayed. In the preferred embodiment, the description of the selected asset is displayed as graphical information in the description window 138 (FIG. 5) of the asset management GUI screen. From step 332, the sub-process proceeds to step 334 in which an interface is provided, the interface allowing the user to select from copy/delete options including "copying" the selected asset, and "deleting" the selected asset. Also in step 334, user input indicating a selected one of the copy/delete options is received. In the preferred embodiment, the user may select the option of copying selected assets by activating the copy button 142 (FIG. 5) of the asset management screen, or the user may select the delete selected assets option by activating the delete button 144 (FIG. 5) of the asset management GUI screen. The delete option is only possible if the selected location, selected via the drop down list 132 (FIG. 5), is a media server. As mentioned above, the execution of the delete action does not actually delete the selected asset from the selected location, but rather removes the selected asset from the catalog of the corresponding media streamer application 20 (FIG. 1). The user of the asset management and scheduling GUI generally does not have authority to actually delete assets. Generally, only a local administrator of the corresponding server may actually delete files.

From step 334, the sub-process proceeds to 336 at which it is determined whether the delete assets option has been selected, and if so, the GUI process proceeds to "F1" (to FIG. 10B) to execute a sub-process for deleting a selected asset from a selected location as described below. Alternatively, if it is determined at 336 that the delete assets option has not been selected, the sub-process proceeds to 338 at which it is determined whether the copy option has been selected, and if so, the process proceeds to "F2" (to FIG. 10C). If it is determined at 336 and 338 that neither the copy option nor the delete option has been selected, the process proceeds back to "F'" (back to FIG. 6).

FIG. 10B shows a flow diagram illustrating a sub-process at 342 for defining a delete action for "deleting" a selected asset from a selected location. The sub-process 342 proceeds from "F1" (from FIG. 10A) to step 344 in which a delete action, associated with instructions for deleting the currently selected asset from the selected location, is defined. From step 344, the sub-process proceeds to step 346 in which the delete action defined in step 344 is added to an action list. In step 348, an updated action list is displayed after which the process proceeds back to "F1'" (back to FIG. 10A). In the preferred embodiment, the sub-process 342 is implemented using the asset management GUI screen 130 (FIG. 5) wherein the user may delete a selected asset, which has been selected via the assets window 136 (FIG. 5), by activating the delete button 144 (FIG. 5) of the asset management GUI screen 130, and then activating the define button 158 (FIG. 5) which adds the defined delete action to the action list box 156 (FIG. 5).

FIG. 10C shows a flow diagram illustrating a sub-process at 350 for "copying" a selected asset from a selected source location to a selected destination location. The sub-process 350 proceeds from "F2" (from FIG. 10A) to step 352 in which the user is prompted to select a destination location. Also in step 352, user input indicating the selected destination location is received, the source location having been selected in step 322 (FIG. 10A). In the preferred embodiment,

sub-process 350 is implemented using the asset management GUI screen 130 (FIG. 5) wherein the user selects the destination location from the list of possible destination locations listed in the "copy-to" window 146 (FIG. 5) as explained above.

Note that upon selecting a destination location, if the user has not already logged on to the server associated with the selected destination location, the asset management and scheduling process executes instructions at the user terminal 12 (FIG. 1) for accessing the selected server associated with the selected destination location (e.g., SERVER_1 or SERVER_2 of FIG. 1). At this point, the user may be required to enter a log-in name and password in order to establish a connection with the selected destination server. A connection with the selected destination server is required in order to verify that the user has authority to copy media data to the destination media server.

From step 352, the sub-process proceeds to step 354 in which an interface is provided, the interface allowing the user to select from timing options including an immediate copying option, and a delayed copying option wherein selected assets are scheduled for copying at a specified later time. Also in step 354, user input indicating the selected timing option is received. From step 354, the sub-process proceeds to 356 at which it is determined whether the timing option of copying a selected asset at a specified later time has been selected. In the preferred embodiment, the user may select displayed copying option by checking the "copy assets later" box 148 (FIG. 5).

If the delayed copying option is selected, the sub-process proceeds from 356 to step 358 in which an interface allowing the user to select a start date and start time is provided. Also at step 358, user input including scheduling information associated with the copy action is received, the scheduling information including a selected start date and start time. In the preferred embodiment, the user selects the start date and start time using the start date drop down window 150 (FIG. 5) and

the start time spin button 152 (FIG. 5) respectively. If the delayed copying option is not selected, then it is assumed that scheduling information associated with the defined copy action specified that the copy action is to be executed presently.

From step 358, the sub-process proceeds to step 360 in which an interface, allowing the user to select an option of renaming the asset before copying, is provided. Also in step 360, user input indicating a selected renaming option is received. In the preferred embodiment, the user may select the option of renaming assets before copying by checking box 154 (FIG. 5) of the asset management GUI screen 130 (FIG. 5). From step 360, the process proceeds to "F3" (to FIG. 10D) to execute further steps of the copy/delete selected assets sub-process.

FIG. 10D shows a flow diagram at 362 illustrating further steps of the copy/delete selected assets sub-process 320 (FIG. 10A). Sub-process 362 proceeds from "F3" (from FIG. 10C) to 364 at which it is determined whether the renaming option has been selected, and if so, the sub-process proceeds to step 366 in which an interface allowing the user to enter a target name for the copied asset is provided. Also in step 366, user input indicating a target name is received. In the preferred embodiment, the user provides input indicating a target name for the selected asset using a rename asset GUI screen 390 shown in FIG. 11. With reference to FIG. 11, the rename asset GUI screen 390 includes a current name window 392 for displaying a current name of the selected asset; a "rename as" window 394 allowing for the user to enter a target name for the selected asset; an OK button 396; a cancel button 398; and an help button 400.

Referring back to FIG. 10D, the sub-process proceeds from 366 to step 368 in which a copy action is defined, the copy action being associated with instructions for copying the currently selected asset, or assets, from the selected source location to the selected destination location in accordance with the specified scheduling information (at the selected date and time if the delayed

copying option has been selected, or presently if the delayed copy option has not been selected) in accordance with the selected renaming option. In the preferred embodiment, the copy action is defined using the GUI components of the asset management GUI screen 130 (FIG. 5) as explained above.

From step 368, the sub-process proceeds to step 370 in which the copy action defined in step 368 is added to an action list, and then to step 372 in which an updated action list is displayed. In the preferred embodiment, copy actions are added to the action list by activating the define button 155 (FIG. 5) of the asset management GUI screen which adds the defined copy action to the action list displayed in window 156 (FIG. 5). In step 374, an interface allowing the user to execute the actions listed in the action list is provided, and user input is received. In the preferred embodiment, the user may execute the actions listed in the action list 156 (FIG. 5) of the asset management GUI screen by activating the OK button 168 (FIG. 5), or the apply button 172 (FIG. 5), after actions have been defined.

From step 374, the sub-process proceeds to 376 at which it is determined whether the option of executing the defined actions has been selected. It is important that all copy actions associated with a particular asset be executed before any delete actions associated with the particular asset so that the particular asset is not deleted before it may be copied. If it is determined at 376 that the defined actions are to be executed, the sub-process proceeds to step 380 in which, for each listed copy action, a set of copy commands and associated parameters is generated, each set of copy commands and associated parameters specifying that the corresponding selected asset is to be "copied" from the corresponding selected source location to the corresponding selected destination location in accordance with the corresponding selected renaming option and in accordance with the specified scheduling information. From step 380, the sub-process proceeds to step 382 in which, for each listed delete action, a set of delete commands and associated parameters is generated, each

set of delete commands and associated parameters specifying that the corresponding selected asset is to be "deleted" from the corresponding selected source location as defined by the corresponding delete action.

From step 382, the sub-process proceeds to step 384 in which each set of copy commands and associated parameters, and each set of delete commands and associated parameters is transmitted to the corresponding media server (e.g., SERVER_1 or SERVER_2 of FIG. 1) which is identified by the corresponding source location. In response to the sets of commands and associated parameters, the media streaming application 20 (FIG. 1) of the corresponding media server is operative to schedule execution of, or presently execute, the corresponding copy and delete operations as defined by the actions specified by the commands and associated parameters. For example, the media streamer application 20 (FIG. 1) of the media server associated with the selected source location is operative to copy the corresponding selected assets from the corresponding selected source location to the corresponding selected destination location in accordance with the selected renaming option and the specified scheduling information. In one embodiment, the asset management and scheduling GUI process, executed at the administrator terminal, transmits the sets of commands and associated parameters to the corresponding media servers via the IP network 16 (FIG. 1). Note that connections with the selected source locations and destination locations were previously made at which time user names and passwords were provided, if required. The connections with the locations need not be continuously maintained but may be re-established when required. From step 384, the process proceeds back to "F'" (back to FIG. 6).

FIG. 12 shows a block diagram illustrating an encoding GUI screen 420 for encoding a new media asset as further explained below. The GUI screen 420 is displayed to the user at the administrative terminal 12 (FIG. 1) when the "encode a new asset icon" 104 (FIG. 3) is selected. The GUI screen 420 includes: an encoder drop down list 422 allowing the user to select an encoder

for encoding a new asset (e.g., SERVER_1 which is connected to camera 24 of FIG. 1); an encoder time display box 424 for displaying the local time at the selected encoder which is a selected one of the media servers; an encoder attributes button 426 for selecting audio and video encoding attributes for the encoding process as further explained below; a preview button 428 for activating a browser plug-in at the administrator terminal 12 (FIG. 1) for displaying a scene to be encoded by the selected encoder; and a duration window 430 for defining schedule information specifying a schedule for the encoding operation as further explained below, the GUI screen 420 also provides for the user to define and schedule streaming operations, recording operations, and playback operations to be performed on the newly encoded asset.

The duration window 430 includes: a manual option button 432 for specifying manual mode operation which allows for the user to manually start and stop encoding as further explained below; and a timed encoding option button 434 for specifying scheduling information associated with the encoding operation, the scheduling information indicating when the encoding operation is to start and stop. The duration window 430 also includes GUI components associated with the timed encoding mode of operation, the components including: a start group box 435 having a start-now button 436 for specifying that encoding is to start immediately, and a schedule time button 438 which enables a start date drop down date-picker 440 and a start time spin button 442 further explained below; and a stop group box 443 having a duration button 444 which enables a duration spin button and window 446, and a schedule time button 448 which enables a start date drop down window 450, and a start time spin button 452, further explained below. Note that when the manual option button 432 is activated, the contents of the start and stop group boxes 435 and 443 are disabled. Also, activating the duration button 444 disables the start date and start time GUI components 450 and 452. When the timed encoding button 434 is activated, the user is prompted to provide scheduling information via the start and stop group boxes 435 and 443. The timed encoding may be scheduled to begin immediately by activating the start now button 436, or at a

scheduled date and time by selecting via components 440 and 442. The timed encoding may be scheduled to stop either after a specified duration selected via the duration window 446, or at a specified time and date selected via the GUI components 450 and 452.

The encoding GUI screen 420 further includes: a streaming check box 454 for specifying that the asset is to be streamed from the encoder to one or more selected end users in real time that is, within a minimal time delay after which the asset is encoded; and a stream box 455 having interface components which are enabled upon checking box 454. The stream box 455, which allows for the user to specify destination end user terminals to which newly encoded asset is to be streamed, includes: a manual option button which enables the user to specify one destination IP address and port number via an IP address text box 462, and a port number text box 464; a predefined option button 458 which enables the user to select from predefined groups of destination IP addresses and port numbers listed in a drop down list 466, the option button 458 also enabling a define button 468 which allows the user define a new group of destination IP addresses and port numbers as further explained below; and a session director requester (SDR) option button 460 which is enabled if the user has third party software. In response to the user providing a start time and duration, SDR determines an available IP address and port number.

The encoding GUI screen 420 further includes: a record asset check box 470 for specifying that an asset is to be recorded as it is encoded; and a record asset box 471 having interface components which are enabled upon checking box 470. The record asset box 471 includes: a "record to" drop down combination box 472 allowing the user to select a record path from previously defined record paths, the selected record path specifying a selected "record to" memory device and directory to which the asset is to be stored; a browse button 474 which displays the "browse local disks" GUI screen 230 (FIG. 7C) for defining a path for a local disk and directory, such as for the local disk 22 of SERVER_1 (FIG. 1); an asset name text box 476 providing for the user to enter text

indicative of a name (e.g., "Trial.mpg") for the new asset; a description text box 478 providing for the user to enter textual information describing the new asset; and three overwrite option buttons explained below.

The three file option buttons include: a prompt option button 480 which, when activated, specifies that the user is to be prompted if an asset having the name indicated in the asset name text box 476 already exists at the selected "record to " memory device; an overwrite option button 482 which, when activated, specifies that any asset having the same name as that indicated in the asset name text box 476 which exists a the "record to" memory device is to be overwritten with the new encoded asset; and an append option button 484 which, when activated, specifies that any asset having the same name as that indicated in the asset name text box 476 is to be appended with the encoded and parsed media data comprising the new asset.

The encoding GUI screen 420 further includes: a schedule playback check box 486, enabled if the manual option button 432 is not selected for scheduling a play-back (streaming or multicasting) operation to be executed on the newly encoded asset in accordance with a play-back schedule. The check box 486 enables a schedule play-back button 488 for activating GUI components for specifying a play-back schedule as further explained below; an OK button 490; a cancel button 492; and a help button 494.

FIG. 13A shows a flow diagram illustrating an encoding sub-process at 500. The depicted sub-process proceeds from "B" (from FIG. 4) to step 502 in which an interface, allowing the user to select a source encoder, is provided. Also in step 502, user input indicating a selected source encoder is received. In the preferred embodiment of the present invention, the encoding sub-process 500 is implemented using the encoding GUI screen 420 (FIG. 12). From step 502, the sub-process proceeds to step 504 in which graphical information indicative of the selected source encoder is

displayed, and local time at the location of the selected source encoder is also displayed. In the preferred embodiment, steps 502 and 504 are implemented using the encoder drop down list 422 (FIG. 12), and the server time window 424 (FIG. 12).

From step 504, the depicted sub-process proceeds to step 506 in which an interface allowing the user to select the option of changing attributes of the selected encoder is provided, and user input indicative of the selected option is received. In the preferred embodiment, step 506 is implemented using the encoder attributes button 426 (FIG. 12) of the encoding GUI screen. From step 506, the sub-process proceeds to 508 at which it is determined whether the option of changing attributes of the selected encoder has been selected, and if so, the process proceeds to "B1" (to FIG. 15) to execute a sub-process of defining encoder attributes.

If it is determined at 508 that the option of changing encoder attributes has not been selected, the sub-process proceeds to step 510 in which an interface allowing user to select a preview option, for previewing a scene to be encoded, is provided. Also in step 510, user input indicative of the selected preview option is received. In the preferred embodiment, step 510 is implemented using the preview button 428 (FIG. 12) of the encoding GUI screen. From step 510, the sub-process proceeds to step 512 at which it is determined whether the preview option has been selected, and if so, the sub-process proceeds to step 513 in which a browser plug-in, showing a scene to be encoded, is displayed. The browser plug-in may be implemented in accordance with any of the variety of well known methods. In step 513, preview commands are transmitted from the administrator terminal 12 (FIG. 1) to the selected encode via the IP network. In response to the preview commands, the selected encoder generates media data associated with a scene to be encoded, and streams the media data back to the administrator terminal. For example, if the selected encoder specifies SERVER_1 (FIG. 1) and camera 24 (FIG. 1), then the camera 24 is activated by SERVER_1 in response to preview commands sent to SERVER_1, and media data is generated by camera 24 (FIG. 1) and

transmitted to the administrator terminal 12 (FIG. 1) at which the media data is displayed to the user in a plug-in displayed within a browser window on the display unit 62 (FIG. 2) of the administrator terminal.

From step 513, the sub-process proceeds to step 514 in which an interface allowing the user to select from encoding options is provided. The encoding options include a manual encoding option, and a timed encoding option. Also in step 514, user input indicating a selected encoding option is received. In the preferred embodiment, step 514 is implemented using the manual option button 432 (FIG. 12) and the timed encoding option button 434 (FIG. 12) of the encoding GUI screen. From step 514, the sub-process proceeds to 516 at which it is determined whether the manual encoding option has been selected, and if so, the process proceeds to "B2" (to FIG. 17). Alternatively, if it is determined at 516 that the manual encoding option encoding has not been selected, the sub-process proceeds to step 518 in which an interface, allowing the user to specify scheduling information for timed encoding is provided. Also in step 518, user input including scheduling information for a timed encoding operation is received. In the preferred embodiment, step 518 is implemented using the GUI components of the start group box 435 (FIG. 12) and the stop group box 443 (FIG. 12) of the encoding GUI screen as described above.

FIG. 13B shows a flow diagram illustrating further steps of the encoding sub-process 500 (FIG. 13A). The sub-process proceeds from "B3" (from FIG. 13A) to step 526 in which: an interface allowing the user to select a real time streaming option is provided; and user input indicative of the selected streaming art option is received. In the preferred embodiment, step 526 is implemented using the streaming check box 454 (FIG. 12) of the encoding GUI screen. From step 526, the sub-process proceeds to step 528 at which it is determined whether the real time streaming option has been selected, and if so, the process proceeds to "B4" (to FIG. 18) to execute a sub-process of defining destination locations to which the encoded asset is to be streamed in real time.

Alternatively, if it is determined at 528 that the real time streaming option has not been selected, the sub-process proceeds to step 530 in which: an interface allowing the user to select a record asset option is provided; and also user input, indicative of whether the record asset option has been selected, is received. In the preferred embodiment, step 530 is implemented using the record asset option check box 470 (FIG. 12) of the encoding GUI screen. From step 530, the sub-process proceeds to 532 at which it is determined whether or not the recording option has been selected, and if so, the process proceeds to "B5" (to FIG. 21) to execute a sub-process of specifying a location to which the newly encoded asset is to be stored, as well as providing other recording options including naming the encoded asset, providing a description of the encoded asset, and specifying overwrite options for recording the asset, all of which are described below. Alternatively, if it determined at 532 that the record asset option has not been selected, the process proceeds to step 536 in which commands and parameters associated with scheduled encoding operations, streaming operations (if specified), and recording operations (if specified) are generated. Scheduled encoding commands and associated parameters are generated based on encoding operation information and encoding scheduling information specified by the user via the encoding GUI components 422 through 452 (FIG. 12) of the encoding GUI screen. Streaming commands and associated parameters are generated based on real time streaming operation information specified by the user via the GUI components of the streaming group box 455 (FIG. 12) of the encoding GUI screen. Recording commands and associated parameters are generated based on recording operation information specified by the user via the GUI components 470 through 484 (FIG. 12) of the encoding GUI screen.

From step 536, the process proceeds to step 538 in which the scheduled encoding commands and associated parameters (as well as the streaming and recording commands and associated parameters, if applicable) are transmitted to the selected source encoder. In response to the received commands, the selected source encoder, which is a selected one of the media servers 18, 30 (FIG.

1) executes the specified scheduled encoding operations (as well as the recording and real time streaming operation, if selected, in accordance with the specified operation information and schedule information. From step 538, the process proceeds to "B'" (back to FIG. 4).

FIG. 14 shows a block diagram illustrating an encoder attributes defining GUI screen 550 in accordance with the present invention. The screen 550 is displayed upon activation of the encoder attributes button 426 (FIG. 12) of the encoding GUI screen 420 (FIG. 12). The GUI screen 550 includes: a logical port name drop down list 552 allowing the user to select an alias name for the selected encoder; an encoder type display window for displaying graphical information indicative of a type of the selected encoder, the window 554 being non-editable; an encoder quality drop down list 556 allowing the user to select an encoding quality such as MPEG1, MPEG2, and other video encoding types; and a set bit rate box 557 for specifying a bit rate for an encoding operation.

The set bit rate box 557 includes: a mux rate option button 558 which enables a low bit rate option button 564 (0.8 Mbps), a medium bit rate option button 566 (1.5 Mbps), and a high bit rate option button 568 (2.2 Mbps); and a video and audio option button 560 which enables a video label and text box 570, and an audio drop down list 572.

The VideoCharger media server offers a solution for both the Internet environments as well as the Intranet. For the Internet clients, especially the home Internet users which are typically connected via slower network connections, VideoCharger will support the delivery of Low Bit Rate video (LBR). The LBR video is based on the Industry Standard H.263 video and G.723 audio from the video conferencing industry. This technology allows audio and video to be served to home Internet users connected with 28.8 Kb modems. Using ~ 16 Kb/s, the LBR video will offer 8 KHz 16 bit PCM audio and 160X120 video at 7.5 frames per second. The LBR video can be encoded at higher quality rates to provide higher resolution or more frames per second for those clients which

are connected via ISDN modems, cable modems or an Intranet network. For the Intranet environment, the VideoCharger media server provides support for higher quality (and higher bit rate) videos. In this environment, both MPEG-1 and MPEG-2 content can be supported at rates up to 8 Mb/s. Near CD quality single-channel audio is supported at 24KHz. This flexible support for higher quality video will allow a multitude of applications to be enhanced with video in the Intranet environment. VideoCharger includes transcoders that can convert either Windows .AVI or .WAV files into low bit-rate format. The compressed data is stored in a unique file format (.IBA) optimized for UDP protocol.

The GUI screen 550 further includes: an advanced box 574 having a mux mode drop down list 576 for specifying how to combine video and audio data, a video source drop down list 578, and a group of pictures (GOP) spin button 580 providing for the user to select from groups of pictures between headers. The GUI screen 550 further includes: an OK button 582 for making everything okay; a cancel button 584; and a help button 586.

FIG. 15 shows a flow diagram illustrating a sub-process for defining encoder attributes at 600. The depicted sub-process proceeds from "B1" (from FIG. 13A) to step 602 in which an interface, allowing the user to select from a list of alias names (logical port names) for the selected encoder is provided. Also in step 602, user input indicated of the selected alias name is received. In the preferred embodiment, the sub-process 600 is implemented using the encoder attributes GUI screen 550 (FIG. 12). Specifically, step 602 is implemented using the logical port name drop down list 552 (FIG. 12). From step 602, the sub-process proceeds to step 604 in which an interface, allowing the user to select from the list of quality types for the selected encoder, is provided. Also in step 604, user input indicative of a selected quality type is received. In the preferred embodiment, step 604 is implemented using the encoder quality drop down list 556 (FIG. 14).

From step 604, the sub-process proceeds to step 606 in which an interface, allowing a user to select from bit rate options including a mux rate option and a video/audio option, is provided. Also in step 606, user input indicative of a selected bit rate option is received. In the preferred embodiment, step 606 is implemented using the set bit rate box 557 (FIG. 14) in which the user may select the mux rate option button 558 or the video/audio option button 560 (FIG. 14). From step 606, the sub-process proceeds to step 608 at which it is determined whether the mux rate option has been selected, and if so, the sub-process proceeds to step 610 in which an interface allowing the user to select a mux rate, is provided. Also in step 610, user input indicative of the selected mux rate is received. In the preferred embodiment, step 610 is implemented using the low, medium, and high buttons 564, 566, and 568 (FIG. 14) of the GUI screen 550.

Alternatively, if it is determined at 608 that the mux rate option has not been selected, the sub-process proceeds directly to step 612 at which it is determined whether the video/audio option has been selected. If the video/audio option has been selected, the sub-process proceeds to step 614 in which an interface, allowing the user to enter a video bit rate and an audio bit rate value is provided. Also in step 614, user input indicative of a selected video bit rate and a selected audio bit rate is received. In the preferred embodiment, step 614 is implemented using the video rate text box 570 and the audio rate drop down list 572 (FIG. 14). From step 614, the process proceeds to "B1'" (back to FIG. 13A).

FIG. 16 shows a block diagram illustrating a GUI screen 620 for manually encoding a new asset. The GUI screen 620 is activated, and displayed upon selection of the manual option button 432 (FIG. 12) of the encoding asset GUI screen. The manual encoding GUI screen 620 provides an interface allowing for the user to remotely start, pause, and stop encoding of video data at the selected encoder, selected via the encoder drop down list 422 (FIG. 12). In accordance with the GUI process of the present invention, manual encoding commands and associated parameters are

generated at the administrator terminal 12 (FIG. 1) and transmitted to the selected encoder via the IP network 16 (FIG. 1). In response to the manual encoding commands, the media streamer application 20 (FIG. 1) of the encoder is operative to start, stop, and pause encoding of media data received from the video camera.

The manual encoding GUI screen 620 includes: a record button 622 for starting encoding of the asset; a pause button 624 for pausing the encoding of the asset; a stop button 626 for stopping the encoding of the asset; an elapsed time window 628 for displaying an elapsed time clock which is running while the asset is being encoded, the elapsed time clock being started, paused, and stopped upon activation of buttons 622, 624, and 626 respectively; a "stream to" window 630 for displaying an IP address and port number to which the encoded asset is being streamed in real time, if applicable; a "record to" window 632 for displaying a path of a location to which the encoded asset is being is recorded to, if applicable; an asset name window 634 for displaying an asset name specified for the encoded asset; a description window 636 for displaying a description of the encoded asset; and an encoder window 638 for displaying a name indicative of the selected encoder.

FIG. 17 shows a flow diagram illustrating a sub-process at 50 of manually and remotely controlling a process of encoding an asset. The sub-process 50 proceeds from "B2" (from FIG. 13A) to step 652 in which a browser plug-in showing a scene to be encoded is displayed. In the preferred embodiment, the sub-process 650 is implemented using the manual encoding GUI screen 620 (FIG. 16) which is displayed in a browser window displayed on the display device of the administrator terminal 12 (FIG. 1).

From step 652, the sub-process proceeds to step 654 in which text indicating details is displayed. The displayed details include an elapsed time, a destination IP address and port number to which the encoded data is to be streamed in real time, a record storage path location for storing

the encoded asset, the asset name, an asset description, and a selected encoder. In the preferred embodiment, step 654 is implemented using the GUI screen 620 which displays the details 628 through 638 (FIG. 16). From step 654, the sub-process proceeds to step 656 in which a user interface, allowing a user to select from manual encoding commands including start/resume encoding, pause encoding, and stop encoding is provided. Also in step 656, a user input indicative of a selected one of the manual encoding commands is received.

From step 656, the sub-process proceeds to 658 at which it is determined whether the start manual encoding option has been selected, and if so, the sub-process proceeds to step 660 in which commands for starting encoding of video data are generated and communicated from the administrator terminal 12 (FIG. 1) to the selected encoder via the IP network (FIG. 1). From step 660, the sub-process proceeds to step 662 in which an elapsed time clock is started . In the preferred embodiment, the elapsed time is displayed in the elapsed time window 628 of the manual encoder GUI screen 620 (FIG. 16).

If it is determined at 658 that the start encoding option has not been selected, the sub-process proceeds to 664 at which it is determined whether the pause encoding option has been selected. If it is determined at 664 that the pause encoding option has been selected, the sub-process proceeds to step 668 in which commands for pausing the process of encoding video data are generated and communicated from the administrator terminal 12 (FIG. 1) to the selected encoder via the IP network (FIG. 1). From step 668, the sub-process proceeds to step 670 in which the elapsed time clock is paused .

If it is determined at 664 that the pause encoding option has not been selected, the sub-process proceeds to 672 at which it is determined whether the stop encoding option has been selected. If it is determined at 672 that the stop encoding option has been selected, the sub-process

proceeds to step 674 in which commands for stopping the encoding of video data are generated and communicated from the administrator terminal 12 (FIG. 1) to the selected encoder via the IP network (FIG. 1). From step 674, the sub-process proceeds to step 676 in which the elapsed time clock is stopped .

FIG. 18 shows a flow diagram illustrating a streaming sub-process at 700 in accordance with the present invention. The streaming sub-process 700 proceeds from "B4" (from FIG. 13B) to step 702 in which an interface, allowing the user to select from streaming destination specification options is provided. The destination specification options include a manual option, a predefined option, a define option, and a session director request option. Also in step 702, user input indicative of the selected destination specification option is received. In the preferred embodiment, step 702 is implemented using the IP address window 462 (FIG. 12) and the port number drop down list 464 (FIG. 12). In the preferred embodiment, the sub-process 700 is implemented using the encoding GUI screen 420 (FIG. 12), and in particular, the GUI components of the stream out group box 455 which is enabled by the stream out check box 454 (FIG. 12). The manual option, predefined option, define option, and SDR option may be selected by the user using the manual option button 456, predefined option button 458, define button 468, and SDR button 460 (FIG. 12) respectively.

From step 702, the depicted sub-process proceeds to 704 at which it is determined whether the manual option has been selected, and if so, the sub-process proceeds to step 706 in which an interface, allowing the user to select a destination IP address and port number, is provided. Also in step 706, user input indicative of the selected destination IP address and selected destination port number is received. From step 706, the sub-process proceeds to 708 at which it is determined whether the predefined option has been selected. Also if it is determined at 704 that the manual option has not been selected, the sub-process proceeds directly from 704 to 708. If it is determined at 708 that the predefined option has been selected, the sub-process proceeds to step 710 in which

a user interface, allowing the user to select a group from a list of predefined groups of destination IP addresses and port numbers, is provided. Also in step 710, user input indicative of selected groups is received. In the preferred embodiment, step 710 is implemented using the GUI component 466 (FIG. 12) described above.

From step 710, and also from 708 in the case that the predefined option has not been selected, the sub-process proceeds to 712 at which it is determined whether the define option has been selected, and if so, the sub-process proceeds to "B41" (to FIG. 20). If is determined at 712 that the define option has not been selected, the sub-process proceeds to 714 at which it is determined whether or not the SDR option has been selected, and if so, the process proceeds to "B42" (to FIG. ?). Alternatively, if it is determined at 714 that the SDR option has not been selected, the process proceeds to step 715 in which session director request software is executed after which the process proceeds to "B4(" (back to FIG. 12B).

FIG. 19 shows a block diagram illustrating a GUI screen 720 for defining a destination group of IP addresses and port numbers. The define group GUI screen 720 includes: a group name text box 722 providing for the user to enter a name for set of IP addresses and port numbers; a delete button 724 which is enabled if there are previously defined groups; and a define group box 725.

The define group box 725 includes: a destination group box 726 having an IP address text box 727 providing for the user to enter one destination IP address, the text box 727 enabling an add button as further explained below, and a port number text box 728 having a spin button providing for the user to select a destination port number, the spin button allowing the user to select values limited to a valid range of port numbers reserved for a multicast operation; an add button 730 for adding the IP address and port number specified in text boxes 727 and 728 to a member list, the add

button resetting the IP address text box 727 to be blank, thereby disabling the add button; a remove button 732 allowing the user to remove selected members from the member list; a member list box 734 for listing IP addresses and port numbers for each member in a format "123.123.123.123.123;5001"; an OK button 736 for adding a group of IP addresses and port numbers to the list of predefined groups displayed in the drop down list 466; a cancel button 738; and a help button 740. The member list box 734 is multi-selectable (shift-contiguous selection; control-disjoint selection). The remove button 732 removes the selected row of IP addresses and port number from the number list. If the member list is empty, the remove button 732 is disabled.

FIG. 20 shows a flow diagram illustrating a sub-process at 750 for defining a destination group of IP addresses and port numbers for the purpose of defining a streaming operation. The sub-process 750 proceeds from "B41" (from FIG. 18) to step 752 in which a user interface allowing the user to enter a destination group name for a set of IP addresses and port numbers is provided. Also in step 752, user input indicating a selected destination group name is received. In the preferred embodiment, the sub-process 750 is implemented using the define group GUI screen 720 (FIG. 19). Step 752 is implemented using the group name text box 722 (FIG. 19). From step 752, the sub-process proceeds to step 754 in which an interface allowing the user to select from a list of previously defined destination group names for the purpose of deleting a selected group name, is provided. Also in step 754, user input indicative of a selected destination group name is received. In the preferred embodiment, step 754 is implemented using the delete button 724 (FIG. 19).

From step 754, the sub-process 750 proceeds to 756 at which it is determined whether the delete option has been selected, and if so, the sub-process proceeds to step 758 in which the selected destination group name is deleted from a member list. From step 758, the sub-process proceeds to step 760 in which a member list of the selected destination group is displayed, the list including at least one group number defined by a corresponding destination IP address and port number. In the

preferred embodiment, the member list is displayed in the member list box 734 (FIG. 19) of the GUI screen for defining groups. From step 760, the sub-process proceeds to step 762 in which an interface, allowing a user to add and remove group members in the selected destination group, is provided. Also in step 762, user input indicative of selected added group members, and selected removed group members is received. In the preferred embodiment, step 762 is implemented using the add and remove button 730 and 732 (FIG. 19) of the GUI screen for defining groups. From step 762, the sub-process proceeds to step 764 in which the member list for the selected destination group is updated. In the preferred embodiment, step 764 is implemented by revising the member list that is displayed in the member list box 734 (FIG. 19). From step 764, the sub-process proceeds to "B41'" (back to FIG. 18).

FIG. 21 shows a flow diagram illustrating a sub-process 780 of recording a new asset. The depicted sub-process proceeds from "B5" (from FIG. 13B) to step 782 in which an interface allowing users to select a record path, associated a "record-to" storage unit, is provided, the path indicating a memory storage location for storing the new asset, that is the asset to be encoded and recorded. Also in step 782, user input indicative of a selected record path is received. In the preferred embodiment, the sub-process 780 is implemented using the encoding GUI screen 420 (FIG. 12). Specifically, step 782 is implemented using the "record-to" text box and drop down list 472 of the record asset group box 471 (FIG. 12). From step 782, the sub-process proceeds to step 784 in which an interface prompting the user to enter an asset name for new asset is provided. Also in step 784, user input indicative of a selected asset name is received. In the preferred embodiment, step 784 is implemented using the asset name text box 476 (FIG. 12) of the encoding GUI screen.

From step 784, the sub-process proceeds to step 786 in which an interface prompting the user to enter a description of content to be encoded is provided. Also in step 786 user input indicative of such a description is received. In the preferred embodiment, step 786 is implemented using the

description text box 478 (FIG. 12) which provides for the user to enter a description of the asset to be encoded. From step 786, the sub-process proceeds to step 788 in which an interface prompting the user to select from overwrite options for specifying what to do in the event that the asset name already exists at the record to path, is provided. The override options include a prompt option, an override option, and an append option. Also in step 788, user input indicative of a selected overwrite option is received. In the preferred embodiment, step 788 is implemented using the prompt, overwrite, and append buttons 480, 482, and 484 (FIG. 12) of the encoding GUI screen 420 (FIG. 12).

From step 788, the sub-process proceeds to 790 at which it is determined whether the manual start/stop option was previously selected, and if so, the process proceeds back to "B" (back to FIG. 3). In the preferred embodiment, the manual start and stop option, which is selected using the manual option 432 (FIG. 12), precludes the process from executing the remaining steps 792 and 794 which provide for playing back the newly encoded asset as further explained below.

If it is determined at 790 that the manual option has not been selected, the sub-process proceeds to step 792 in which: an interface, allowing the user to select a schedule playback option, is provided; and user input indicative of the schedule playback option is received. From step 792, the sub-process proceeds to 794 at which it is determined whether the schedule playback option has been selected, and if so, the process proceeds to "B6" (to FIG. 23A). Alternatively, if it is determined at 794 that the scheduled playback option has not been selected, the process proceeds to "B5" (back to FIG. 13B).

FIG. 22 shows a block diagram illustrating a playback scheduling GUI screen at 800 for scheduling a playback, or streaming, operation for a newly encoded asset. The GUI screen 800 is activated and displayed upon selection of the schedule playback option button 486 (FIG. 12) of the encoding GUI screen. The GUI screen 800 provides for scheduling an asset to be played back at a

later time after encoding of the asset as described above with reference to the encoding GUI screen 420 (FIG. 12). The GUI screen 800 includes: an asset name text box 802 which displays the asset name which is also displayed in the asset name text box 476 (FIG. 12) of the encoding GUI screen, the text box 802 being non-editable; a description text box 804 which is also non-editable, and which displays a description of the asset to be recorded, the description matching the one displayed in the description text box 478 (FIG. 12) of the GUI screen; and a schedule list box 806 for displaying a list of schedule entries, each including graphical information indicative of an asset name for an associated asset, a start date and start time for beginning a play-back schedule for the associated asset, a duration and interval of the play-back schedule, and destination information indicative of a destination or group of destination addresses for the corresponding play-back operation.

The GUI screen 800 also includes a define schedule box 808 including: a start group box 810 providing interface components allowing for the user to define a start date and time for beginning a play-back schedule as further explained below; a destination address group box 812 providing interface components allowing for the user to specify one or more destinations to which the asset is to be streamed in accordance with the play-back operation; a duration group box 814 providing interface components allowing for the user to specify a duration and interval of the play-back schedule as further explained below; an enable notification check box 816 enabling a notification button 818 which is used to activate a GUI screen for defining notification options for notifying selected end users, and selected groups of end users, regarding specified events under selected circumstances; an add button 820 for adding schedule entries to the schedule list 806; a cancel button 826; an apply button 828; a reset button 830; and a help button 832.

The start group box 810 of the define schedule box 808 includes: a start date drop down list 840 providing for the user to select a start date for the playback schedule, and a start time spin button 842 for selecting a starting time for the playback schedule. The destination group box 812 includes:

a manual option button 844 which enables an IP address window 846 for entering a destination IP address, and a port number spin button 848 for selecting a destination port number; a predefined option button 850 which enables a predefined drop down list 852 for selecting a predefined group of destination IP addresses and port numbers, and a define button 854 further explained below; and an SDR button 856.

The duration group box 814 includes: a run option button 860 which enables a run time spin button 862 allowing for the user to specify a play-back scheduled defined by a loop count value as further explained below; a repeat option button 864 allowing the user to define the duration of the playback schedule such that the asset is played repeatedly until the schedule entry associated with the asset is removed from the schedule list 806; and an interval option button 866 which enables a specify button 868 for activating another GUI screen for the purpose of specifying an interval playback schedule as further explained below; and a time to live spin button 870 for specifying a number of routers that the stream can pass through.

FIG. 23A shows a flow diagram illustrating a sub-process at 880 for scheduling playback of a new asset. The depicted sub-process proceeds from "B6" (from FIG. 21) to step 882 in which an asset name and description associated with a currently selected asset is displayed. In the preferred embodiment, the sub-process 880 is implemented using the GUI screen 800 (FIG. 22). From step 882, the sub-process proceeds to step 884 in which a schedule list is displayed. The schedule list includes: a list of assets to be played back, each asset having an associated playback start date and start time, an associated playback duration program, and an associated playback destination. In the preferred embodiment, the schedule list is implemented by the schedule list box 806 of the scheduling GUI screen 800 (FIG. 22). From step 884, the sub-process proceeds to step 886 in which an interface, allowing the user to select a playback start date and start time for the selected asset, is provided. Also in step 886, user input indicative of the selected start date and start time is received.

In the preferred embodiment, step 886 is implemented using the start schedule group box 810 (FIG. 22) of the play-back scheduling GUI screen. From step 886, the sub-process proceeds to step 888 in which an interface, allowing the user to select from playback destination specification options for the selected asset, is provided. The destination specification options include: a manual option, a predefined destination option, a define destination option, and a session director requester (SDR) option. Also in step 888, user input indicative of the selected destination specification option is received. In the preferred embodiment, step 888 is implemented using the destination specification group box 812 (FIG. 22) of the play-back scheduling GUI screen.

From step 888, the sub-process proceeds to 890 at which it is determined whether the manual destination specification option has been selected, and if so, the sub-process proceeds to step 892 in which an interface, allowing the user to select a destination IP address and destination port number, is provided. Also in step 892, user input indicative of the selected destination IP address and selected port number is received. In the preferred embodiment, step 892 is implemented using the IP address text box 846 and the port number spin button 848 (FIG. 22) of the play-back scheduling GUI screen.

Alternatively, if it is determined at 890 that the manual destination specification option has not been selected, the sub-process proceeds from 890 to 894 at which it is determined whether the predefined destination specification option has been selected, and if so, the sub-process proceeds to step 896 in which an interface, allowing the user to select from predefined groups of destination IP addresses and port numbers, is provided. Also in step 896, user input indicative of the selected group of destination IP addresses and port numbers is received. From step 896, the process proceeds to "B7" (to FIG. 23B).

FIG. 23B shows a flow diagram illustrating further steps of the sub-process 880 (FIG. 23A)

for scheduling playback of an asset. The depicted sub-process proceeds from "B7" (from FIG. 23A) to 902 at which it is determined whether the define option has been selected, and if so, the sub-process proceeds to step 904. In step 904, an interface allowing the user to enter a destination group name for a set of IP addresses and port numbers, is provided. Also in step 904, user input including the selected destination group name is received. In the preferred embodiment, step 904 is implemented using the define groups GUI screen 720 (FIG. 19). From step 904, the sub-process proceeds to step 906 in which an interface, allowing the user to select from a list of previously defined destination group names for the purpose of deleting the selected group name is provided. Also in step 906, user input indicative of the selected destination group name is received. In the preferred embodiment, step 906 is implemented using the remove button 732 of the define groups GUI screen 720 (FIG. 19). From step 906, the sub-process proceeds to 908 at which it is determined whether a selected group name is to be deleted, and is so, the sub-process proceeds to step 910 in which the selected destination group name is deleted.

In step 912, a member list for the selected destination group is displayed. The list includes at least one group member defined by a corresponding destination IP address and port number. In the preferred embodiment, the member list is implemented using the member list box 734 (FIG. 19). From step 912, the sub-process proceeds to step 914 in which an interface, allowing the user to add and remove group members in the selected destination group, is provided. Also in step 914, user input indicative of the selected added group members and selected removed group members is received. In step 916, the member list for the selected destination group is updated. From step 916, and also from 902 if it is determined that the define option has not been selected, the sub-process proceeds to step 918 at which it is determined whether the SDR option has been selected. If it is determined at 918 that the SDR option has been selected, the sub-process proceeds to step 920 to execute the SDR program, after which the process proceeds to "B9" (to FIG. 23C).

FIG. 23C shows a flow diagram illustrating a sub-process at 930 for defining the duration of a play-back schedule for a play-back of an asset. The depicted proceeds from "B9" (from FIG. 23B) to step 932 in which an interface, allowing the user to select from playback schedule duration specification options, is provided. The playback schedule duration specification options include a run option, a repeat option, and a playback interval option. Also in step 932, user input indicative of the selected duration option is received. In the preferred embodiment, step 932 is implemented using the duration box 814 (FIG. 22) of the scheduling GUI screen 800 which includes the run option button 860, the repeat until removed option button 864, and the interval option button 866. From step 932 the sub-process proceeds to step 934 at which it is determined whether the run option has been selected, and if so, the sub-process proceeds to step 936. In step 936, an interface allowing the user to enter a loop count value is provided. The loop count value indicates a number of times that the asset is to be played back. Also in step 936, user input indicative of the selected loop count value is received. In the preferred embodiment, step 936 is implemented using the spin button 862 (FIG. 22) of the scheduling GUI screen 800. In step 938, a duration program is defined based on the selected loop count value selected in step 936.

If it is determined at 934 that the run option has not been selected, the sub-process proceeds from 934 to 940 at which it is determined whether the repeat option has been selected. If it is determined at 940 that the repeat option has been selected, the sub-process proceeds to step 942 in which a playback duration program associated with the selected asset is defined to repeat until removed from the schedule list 806 (FIG. 22).

If it is determined at 940 that the repeat option has not been selected, the sub-process proceeds to 944 at which it is determined whether the interval option has been selected, and if so, the process proceeds to "B10" (to FIG. 25) to execute a sub-process of defining a playback interval for playing back the to be encoded asset. Alternatively, if it is determined at 944 that the interval

option has not been selected, the sub-process proceeds to step 946.

In step 946, an interface allowing the user to select a notification option is provided. Also in step 946, user input indicative of the selected notification option is received. In the preferred embodiment, step 946 is implemented using the enable notification check box 816 (FIG. 22). From step 946, the sub-process proceeds to step 948 at which it is determined whether the notification option has been selected, and if so, the process proceeds to "B11" (to FIG. 27). Alternatively, if it is determined at 948 the notification option has not been selected, the process proceeds to "B14" (to FIG. 23D).

FIG. 23D shows a flow diagram illustrating a sub-process at 950 illustrating further steps of the sub-process 880 (FIG. 23A) for scheduling playback of a new asset. The depicted proceeds from "B14" (from FIG. 23C) to step 952 in which a set of play-back commands and associated parameters is generated for each defined play-back action. Each set of play-back commands and parameters is generated based on: play-back schedule information including the play-back start date and start time defined for the associated play-back action, and the play-back duration program defined for the associated play-back action; and the associated play back-destinations defined for the associated play-back action. From step 952, the sub-process proceeds to step 953 in which the commands and associated parameters associated with each of the play-back actions are transmitted from the administrator terminal 12 (FIG. 1) to the corresponding selected play-back media server for executing a corresponding play-back operation in accordance with the play-back schedule information.

For example, in the preferred embodiment, a set of play-back commands and parameters is transmitted from the administrator terminal to a selected play-back media server associated with the "record-to" location selected using the "record-to" drop down list and window 472 (FIG. 12). The

set of play-back commands and parameters programs the selected encoder to execute play-back operations beginning on the play-back start date and start time specified using the start box 810 (FIG. 22), in accordance with a play-back schedule specified by the user via the GUI components of the duration box 814 (FIG. 22). Each iteration of the play-back operation includes: reading the new asset from the "record-to" location specified using the "record-to" drop down list and window 472 (FIG. 12); and streaming the asset to the play-back destination locations specified by the user via the destination specification box 812 (FIG. 22) of the play-back scheduling GUI screen.

From step 953, the sub-process proceeds to step 954 in which it is again determined whether the notification option has been selected, and if not, the process proceeds to "B5"" (to FIG. 27). In the preferred embodiment, step 954 is implemented using the enable notification check box 816 (FIG. 22). Alternatively, if it is determined at 954 that the notification option has been selected, the sub-process proceeds to step 955 in which a set of notification commands and associated notification parameters is generated for each defined notification action. Each set of notification commands and notification parameters is generated based on user input received in response to GUI screens for defining selected circumstances under which selected notification messages are to be provided to selected end users as further described below with respect to figs. 26 though 30. From step 955, the sub-process proceeds to step 956 in which the notification commands and associated notification parameters associated with each of the play-back actions are transmitted from the administrator terminal 12 (FIG. 1) to the corresponding selected play-back media server for executing a corresponding notification operation.

FIG. 24 shows a block diagram illustrating a GUI screen 960 for defining an interval for a playback schedule. The GUI screen 960 is displayed, and activated, upon activation of the interval option button 866 (FIG. 22). The GUI screen 960 includes: an interval drop down list 962 for selecting an interval from intervals including hourly, daily, weekly, monthly (dates), monthly (days),

and yearly; a frequency drop down list 964 allowing the user to select a value associated with the selected interval (i.e., if the selected interval is weekly, the frequency drop down list allows the user to select one of the seven days of the week); a duration box 966 for defining a time at which the defined interval is to end as further explained below; a current selection text box for displaying the current selection at 968; an OK button 970; a cancel button 972; and a help button 974. The duration box 966 includes: a repeat option button 976 for specifying that the interval is to repeat until removed from the schedule list; and an end date option button 978 which enables a drop down date picker 979 for allowing the user to specify an end date. In the preferred embodiment, the drop down date picker 979 activates a selectable calendar GUI screen as further explained below.

FIG. 25 shows a flow diagram illustrating a sub-process at 980 for defining a playback schedule interval. The sub-process 980 proceeds from "B10" (from FIG. 23C) to step 982 in which an interface, allowing the user to select from interval frequency options, is provided. Also in step 982, user input indicative of the selected frequency option is received. In the preferred embodiment, sub-process 980 is implemented using the define interval GUI screen 960 (FIG. 24). Specifically, step 982 is implemented using the interval drop down list 962 (FIG. 24) and the frequency drop down list 964 (FIG. 24) of the define interval GUI screen. From step 982, the sub-process proceeds to step 984 in which a playback interval is defined based on the interval frequency options selected in step 982. For example, the user may select the weekly interval in the interval drop down list 962 (FIG. 24), and the "every Sunday" frequency using the drop down list 964 (FIG. 24).

From step 984, the sub-process proceeds to step 986 at which an interface, allowing the users to select from duration options, is provided. Also in step 986, user input indicative of the selected duration option is received. In the preferred embodiment, step 986 is implemented using the duration box 966 (FIG. 24) of the defined interval GUI screen 960. From step 986, the sub-process proceeds to step 988 at which it is determined whether the repeat option has been selected. In the

preferred embodiment, the repeat option is selected by activating the repeat option button 976 (FIG. 24). If it is determined at 988 that the repeat option has been selected, the sub-process proceeds to step 990 in which an asset is scheduled to be played back in accordance with the selected frequency option, selected in step 982, until removed from the schedule list.

Alternatively, if it is determined at 988 that the repeat option has not been selected, the sub-process proceeds to step 992 in which it is determined whether the end date option has been selected, and if so, the sub-process proceeds to step 994. In step 994, an interface, allowing the user to select an end date for the scheduled playback, is provided. Also in step 994, user input indicative of the selected end date is received. In the preferred embodiment, the end date is selected by the user using the drop down list 979 (FIG. 24).

From step 994, the sub-process proceeds to step 996 in which an asset to be played back in accordance with the selected frequency option until the selected end date. From step 996, the process proceeds back to "B10'" (back to FIG. 23C).

FIG. 26 shows a block diagram illustrating a GUI screen 1000 for defining notification e-mail messages to be sent to selected persons, and users, in predefined circumstances. The GUI screen 1000 is activated, displayed, upon activating the notification button 818 (FIG. 22) of the asset scheduling GUI screen. The GUI screen 1000 includes: a notification box 1002 having user interface means for defining the circumstances under which e-mail notification is to be sent, specifying destination end users to whom the e-mail messages are to be sent, and defining and editing the contents of the messages to be sent; a notification list window 1004 which displays a list of persons, or end users, to who are to be notified; an add button 1006 for adding notification entries defined using the notification box 1002 to the notification list 1004; a remove button 1008 for removing selected notification entries from the notification list 1004; an OK button 1010; a cancel button

1012; and a apply button 1014; a reset button 1016; and a help button 1018.

The notification box 1002 includes: a notification circumstances option window 1019 having a plurality of option buttons for defining circumstances under which e-mail notification is to be provided to selected persons, the notification option buttons including a delivery fails option button 1020 for specifying an e-mail is to be sent to selected persons when streaming of the asset fails, a deliver succeeds button 1022 specifying that notification are to be sent to the selected persons upon success of (??), and an always notification option button 1024 specifying that e-mail messages are to be sent to the selected persons in the event of failure or success; a drop down list 1026 allowing the user to select who they want an e-mail notification sent to (groups?); a define button 1028 for activating a define groups GUI screen for defining groups of e-mail addresses as further explained below; a current message text box 1030 which is a non-editable for displaying the e-mail message to be sent to the selected destination addresses; and an edit message button 1032 for activating an edit message GUI screen for editing the current message to be sent as further explained below.

FIG. 27 shows a flow diagram illustrating a sub-process at 1050 of providing a user interface allowing a user to define circumstances under which notification e-mail messages are to be sent to selected persons, user interface also allowing the user to draft and edit the e-mail messages and select the destination e-mail addresses to which the messages are to be sent. In the preferred embodiment, the sub-process 1050 is implemented using the GUI screen 1000 (FIG. 26). The depicted sub-process proceeds from "B11" (from FIG. 23C) to step 1052 in which a notification list including previously defined notification entries is displayed. In the preferred embodiment, the notification list is implemented using the notification list box 1004 (FIG. 26) of the define notification GUI screen. From step 1052, the sub-process proceeds to step 1054 in which an interface, allowing the user to select from notification circumstance options, is provided. Also in step 1054 user input indicative of a selected notification circumstance option is received. In the preferred embodiment,

step 1054 is implemented using the notification circumstance option window 1019 (FIG. 26) which provides for the user to select from the options of sending e-mail notification messages when delivery fails, when delivery succeeds, and when delivery fails or succeeds, that is always. From step 1054, the sub-process proceeds to step 1056 in which an interface, allowing the user to select from groups of destination e-mail addresses, is provided. Also in step 1056, user input indicative of a selected group is received. In the preferred embodiment, step 1056 is implemented using the drop down list 1026 (FIG. 26) which provides a list of predefined groups of destination e-mail addresses.

From step 1056, the sub-process proceeds to step 1060 in which an interface, allowing the user to select an option of defining a new group of destination e-mail addresses, is provided. Also in step 1060, user input indicative of the selected option is received. From step 1060, the sub-process proceeds to 1062 at which it is determined whether the option of defining a new group has been selected, and if so, the process proceeds to "B12" (to FIG. 29) to execute a sub-process of defining a group of destination e-mail addresses as further explained below.

Alternatively, if it is determined at 1062 that the option of defining a new group has not been selected, the sub-process proceeds to step 1064 in which a current message to be sent to the selected group is displayed. In the preferred embodiment, the current message is displayed in the current message window 1030 (FIG. 26). From step 1064, the sub-process proceeds to step 1066 in which an interface, allowing the user to select the option of editing the current message, is provided. Also in step 1066 user input indicative of the user's response to the option is received. In the preferred embodiment, the option of editing the messages provided by the edit message box 1032 (FIG. 26). From step 1066, the sub-process proceeds to step 1068 at which it is determined whether the option of editing the message has been selected, and if so, the process proceeds to "B13" (to FIG. 31) to execute a sub-process of providing a user interface for editing the message to be sent. Note that the

current message box 1030 is not editable.

In step 1070, a new notification entry is defined based on the user selected notification circumstances, selected notification destination e-mail addresses, and the contents of the current message. Also in step 1070, the new notification entry is added to the notification list. From step 1070, the sub-process proceeds to step 1072 in which the revised notification list is displayed. In the preferred embodiment, the revised notification list is displayed in the notification list box 1004 (FIG. 26). From step 1072, the process proceeds back to "B11"' (back to FIG. 23C).

FIG. 28 shows a block diagram illustrating a GUI screen 1080 for defining groups of destination e-mail addresses for the purpose of sending notification messages to the destination addresses. The GUI screen 1080 is displayed and activated upon activation of the define button 1028 (FIG. 26) of the define notification GUI screen. The GUI screen 1080 includes: a group name text box 1082 providing for the user to enter a name for a set of e-mail addresses; a delete button 1084 which is enable if there are previously defined groups listed in the drop down list 1082, the delete button providing for deleting a selected group name including the e-mail addresses associated with the selected group name; a define group box 1086 providing user interface means used for defining a set of e-mail addresses associated with the group name displayed in the group name text box 1082.

The defined group box 1086 includes: an e-mail address text box 1090 providing for the user to enter a destination e-mail address; a member list box 1092 for displaying a list of e-mail addresses associated with the group name displayed in the group name text box 1082, the member list box being multi-selectable; an add button 1094 for adding the destination e-mail address listed in the address box 1090 to the member list displayed in the member list box 1092; a remove button 1096 for removing selected ones of the e-mail addresses listed in the member list from the member list;

an OK button 1098; a cancel button 1100; and a help button 1102.

FIG. 29 shows a flow diagram illustrating a sub-process at 1110 for defining a group of destination e-mail addresses for the purpose of sending notification messages to the group. In the preferred embodiment, the sub-process 1110 is implemented using the define groups GUI screen 1080 (FIG. 28). The sub-process 1110 proceeds from "B12" (from FIG. 27) to step 1112 in which an interface, allowing the user to enter a group name for a new group of e-mail addresses, is provided. Also in step 1112, user input indicative of a group name is received. In the preferred embodiment, step 1112 is implemented using the group name text box and drop down list 1082 (FIG. 28). From step 1112, the sub-process proceeds to step 1114 in which the member list including e-mail addresses associated with the selected group name is displayed. In the preferred embodiment, the member list is displayed in the member list box 1092 (FIG. 28). From step 1114, the sub-process proceeds to step 1116 in which an interface, allowing the user to select add and remove options for adding and removing members from the current member list, is provided. Also in step 1116, user input indicative of the selected add/remove option is received. In the preferred embodiment, step 1116 is implemented using the add and remove buttons 1094 and 1096 (FIG. 28) of the define groups GUI screen. From step 1116, the sub-process proceeds to 1118 at which it is determined whether the remove option has been selected, and if so, the sub-process proceeds to step 1120 in which a user interface, allowing the user to select a member from the member list, is provided. Also in step 1120, user input indicative of the selected member is received. In the preferred embodiment, members listed in the member list box 1092 (FIG. 28) are multi-selectable (shift-contiguous; CTRL-disjoint selection). Also in step 1120, user input indicative of a selected member is received.

From step 1120, the sub-process proceeds to step 1122 in which the selected member is removed from the list.

If it is determined at 1118 that the remove option has not been selected, it is assumed that the add option has been selected, and the sub-process proceeds to step 1126 in which a user interface, allowing the user to enter an e-mail address associated with a new member is provided. Also in step 1126 user input indicative of an e-mail address associated with a new member is received. From step 1126, the sub-process proceeds to step 1128 in which the e-mail associated with the new member is added to the member list displayed in the member list box 1092 (FIG. 28). From step 1128, the sub-process proceeds to step 1130 in which the revised member list is displayed. In the preferred embodiment, the revised member list is displayed in the member list box 1092 (FIG. 28) of the define groups GUI screen. From step 1130, the process proceeds back to "B12'" (back to FIG. 27).

FIG. 30 shows a block diagram illustrating an edit message GUI screen 1140. The screen 1140 is activated in response to the user activating the edit message button 1032 (FIG. 26) of the defined notification GUI screen. The edit message GUI screen 1140 includes: header attributes including an asset name check box 1142 for including an asset name in the message; a status attribute check box 1144 for including the status of the message; a start check box 1146 for specifying that the start date and time of a multicast is to be included in the e-mail message; a error code check box 1150 for specifying a predefined error code indicative of a failure in delivery of a multicast; and a destination box 1152 for specifying the destination e-mail address of or associated with the message. The GUI screen 1140 further includes a current message text box 1154 for displaying a current message, the contents of the current message text box being editable by the user; an OK button 1156; a cancel button 1158; and a help button 1160.

FIG. 31 shows a flow diagram illustrating a sub-process at 1170 for editing a message to be sent to selected e-mail destination addresses. The depicted sub-process proceeds from "B13" (from FIG. 27) to step 1172 in which an interface, allowing the user to select from a plurality of header

attributes, is provided. The selected attributes are to be added to the e-mail message. Also in step 1172, user input indicative of selected header attributes are received. In the preferred embodiment, step 1172 is implemented using the header attributes box 1141 of the edit message GUI screen 1140 (FIG. 30).

From step 1172, the sub-process proceeds to step 1174 in which the current message is displayed. In step 1176, an interface allowing the user to edit the current message is provided. Also in step 1176 user input for editing the current is received. In step 1178, the revised current message is displayed. In the preferred embodiment, the steps 1174, 1176, and 1178 are implemented using the current message text box 1154 of the edit message GUI screen 1140 (FIG. 30). From step 1178, the process proceeds to "B13'" (back to FIG. 27).

FIG. 32 shows a block diagram illustrating a GUI screen 1200 which provides an interface for multicasting existing assets. The multicasting GUI screen 1200 is activated upon selection of the multicast option. The GUI screen 1200 includes an asset select box 1202 for selecting an existing asset located at a selected one of the available media servers (i.e., SERVER_1 or SERVER_2 of FIG. 1), the box 1202 including: an available server drop down list 1204 providing for the user to select a media server that contains assets; a server time text box 1206 for displaying the local time at the selected server, the server time being active (the seconds are clicking away in real time); an existing assets list box 1208 for displaying a list of asset names associated with assets located at the selected server, and also displaying the date and time that the corresponding asset was last modified; and a description label box 1210 for displaying graphical information indicative of a description of the selected asset located at the selected server.

The GUI screen 1200 further includes: a define schedule box 1212 providing user interface means for defining a schedule including a start date and time, a duration, and destination end users

associated with the selected asset at the selected media server and also for defining notification messages to be sent to selected persons under selected circumstances as further explained below; a schedule list 1214 for displaying a list of schedule entries each of the schedule entries including an asset name, a start date and time for multicasting the asset, a duration for the scheduled program associated with the asset, and destination locations to which the asset is to be streamed; a remove button 1216 for removing selected ones of the scheduled entries from the schedule list 1214; an OK button 1218; a cancel button 1220; an apply button 1222; a reset button 1224; and a help button 1226.

The define schedule box 1212 is substantially similar to the define schedule box 808 of the playback scheduling GUI screen 800 (FIG. 22), and includes a start box 1230. The start box 1230 includes: a start now option button 1232 for specifying that the selected asset is to be multicasted from the selected server immediately; and a schedule time option button 1234 for specifying a start date and start time for a program for multicasting the selected asset from the selected server. The schedule time option button 1234 enables a start date drop down date picker 1236 allowing the user to select a date via a selectable calendar view as further explained below, and a start time spin button 1238 allowing for the user to select a start time for beginning the mutlicasting schedule.

The GUI screen 1200 further includes: a destination group box 1240 which is identical to, and which includes all of the same interface components as, the destination group box 812 of the playback scheduling GUI screen 800 (FIG. 22); and a duration group box 1240 which is identical to, and which includes all of the same interface components as, the duration group box 814 of the playback scheduling GUI screen 800 (FIG. 22).

An enable notification check box 1244 enables a notification button 1246 which is used to activate a GUI screen for defining notification options for notifying selected end users, and selected

groups of end users, regarding specified events under selected circumstances. Selecting the notification button 1246 causes the GUI process to activate the GUI screen 1000 (FIG. 26) for defining notification e-mail messages to be sent to selected persons, and users, in predefined circumstances. A define button 1248 provides for adding schedule entries to the schedule list 1214.

FIG. 33A shows a flow diagram illustrating a multicasting sub-process at 1250 in accordance with the present invention. The sub-process 1250 proceeds from "C" (from FIG. 4) to step 1252 in which an interface, allowing the user to select from a displayed list of media servers is provided. Also in step 1252, user input indicative of a selected one of a media servers is received. In the preferred embodiment, the sub-process 1250 is implemented using the multicasting GUI screen 1200 (FIG. 32). From step 1252, the sub-process proceeds to step 1254 in which the local time at the selected media server is displayed. In the preferred embodiment, the local time is displayed in the server time window 1206 (FIG. 32) of the multicasting GUI screen.

From step 1254, the sub-process proceeds to step 1256 in which a list of assets located at the selected media server is displayed. Also in step 1256, a corresponding date and time that each of the assets were last modified is displayed. In the preferred embodiment, step 1256 is implemented using the existing assets display box 1208 (FIG. 32) of the multicasting GUI screen. From step 1256, the sub-process proceeds to step 1258 in which a user interface, allowing the user to select from the list of assets located at the selected location, is provided. Also in step 1258, user input indicative of the selected asset is received. From step 1258, the sub-process proceeds to step 1260 in which a description of the selected asset is displayed. In the preferred embodiment, the description of the selected asset is displayed in the description box 1210 (FIG. 32) of the multicasting GUI screen. From step 1260, the sub-process proceeds to step 1262 in which a schedule list, including a list of assets to be played back, is displayed. Each of the listed assets has an associated start date and time, and associates playback duration program, and an associated

playback destination, all of which are displayed along with the schedule list. In the preferred embodiment, step 1262 is implemented using the schedule list display box 1214 (FIG. 32) of the multicasting GUI screen. In step 1264, a user interface, allowing the user to select from multicast start options, is provided.

The multicasting start options include an immediate start option, and a scheduled start option. Also in step 1264, user input indicative of the selected multicast start option is received. In the preferred embodiment, step 1264 is implemented using the start box 1230 (FIG. 32) of the multicasting GUI screen.

From step 1264, the sub-process proceeds to step 1266 at which it is determined whether the immediate start option has been selected. The determination at 1266 is made and the preferred embodiment by determining the "now" option button 1232 (FIG. 32) has been selected. If it is determined at 1266 that the immediate start option has not been selected, the sub-process proceeds to step 1268 at which an interface, allowing the user to select a multicast start date and start time for the selected asset, is provided. Also in step 1268, user input indicative of the selected start date and start time is received. In the preferred embodiment, step 1268 is implemented using the start date drop down list 1236 and the start time spin button 1238 (FIG. 32) of the multicasting GUI screen. Alternatively, if it is determined at 1266 that the immediate start option has been selected, the sub-process proceeds from 1266 to step 1270 in which a multicast start time is defined based on the selected multicast start option. If the immediate start option has been selected, the multicast start time is defined to be a time immediately upon selection of the OK button 1218 (FIG. 32). If the scheduled start option has been selected, the start time is defined to be the selected start date and selected start time. From step 1270, the process proceeds to "C1" (to FIG. 33B).

FIG. 33B shows a flow diagram illustrating a sub-process 1280 for specifying destinations

to which the existing asset is to be multicasted, or streamed. In the preferred embodiment, the sub-process 1280 is implemented using the destination box 1240 (FIG. 32) of the multicasting GUI screen. The sub-process 1280 proceeds from "C1" (from FIG. 33A) to step 1282 in which an interface, allowing the user to select from multicast destination specification options, is provided. The multicast destination specification options include a manual specification option, a predefined option, a defined option, and an SDR option. Also in step 1282, user input indicative of the selected option is received. From step 1282, the sub-process proceeds to step 1284 at which it is determined whether the manual option has been selected. In the preferred embodiment, the manual option is determined to be selected if the manual option button 844 (FIG. 32) has been selected by the user. If it is determined at 1284 that the manual option has been selected, the sub-process proceeds to step 1286 in which an interface, allowing the user to select a destination IP address and destination port number, is provided. Also in step 1286, user input indicative of the selected destination IP address and selected port number is received. In the preferred embodiment, step 1286 is implemented using the IP address text box 846 and the port number spin button 1246 (FIG. 32). Alternatively, if it is determined at 1284 that the manual option has not been selected, the sub-process proceeds to 1288 at which it is determined whether the predefined option has been selected. In the preferred embodiment, the predefined option is determined to be selected if the predefined option button 850 (FIG. 32) has been selected by the user.

If it is determined at 1288 that the predefined option button has been selected, the sub-process 1280 proceeds to step 1290 in which an interface, allowing the user to select from predefined of group of destination specification IP addresses and port numbers, is provided. Also in step 1290, user input indicating a selected group of destination IP addresses and port numbers is received. In the preferred embodiment, step 1290 is implemented using the predefined group drop down list 852 (FIG. 32) of the multicasting GUI screen. From step 1290, the process proceeds to "C2" (to FIG. 33C).

FIG. 33C shows a flow diagram illustrating further steps of the destination specification and multicasting sub-processes. The sub-process 1300 proceeds from "C2" (from FIG. 33B) to 1302 at which it is determined whether the define destination specification option has been selected. In the preferred embodiment, the define option is determined to be selected if the define button 854 (FIG. 32) has been selected by the user.

If it is determined at 1302 that the define option has been selected, the sub-process proceeds to step 1304 in which an interface, allowing the user to enter a destination group name for a corresponding set of IP addresses and port numbers is provided. Also in step 1304, user input indicative of a selected destination group name is received. In the preferred embodiment, step 1304 is implemented using the define groups GUI screen 720 (FIG. 19).

From step 1304, the sub-process proceeds to step 1306 in which an interface, allowing the user to select from a list of previously defined destination group names for the purpose of deleting the selected group name, is provided. Also in step 1306, user input indicative of the selected destination group name is received. In the preferred embodiment, step 1306 is implemented using the member list box 734 (FIG. 19) and the remove button 732 (FIG. 19) of the define group GUI screen. From step 1306, the depicted sub-process proceeds to step 1308 at which it is determined whether a selected group name is to be deleted, and if so, the sub-process proceeds to step 1310 in which the selected destination group name is deleted.

Alternatively, if it is determined at 1308 that a selected destination group name is not to be deleted, the sub-process proceeds to step 1312 in which a member list is displayed for the selected destination group. The member list includes a least one group member defined by a corresponding destination IP address and destination port number. In the preferred embodiment, step 1312 is implement using the member list box 734 (FIG. 19) of the defined groups GUI screen. From step

1312, the sub-process proceeds to step 1314 in which an interface, allowing the user to add and remove group members in the selected destination group, is provided. Also in step 1314 user input indicative of the selected added group members and selected remove group members is received. In the preferred embodiment, step 1314 is implemented using the member list box 734 (FIG. 19) which allows for selecting members from the member list, the add button 730 (FIG. 19), and a remove button 732 (FIG. 19).

From step 1314, the sub-process proceeds to step 1316 in which a member list associated with the selected destination group is updated. From step 1316, the sub-process proceeds to step 1318 at which it is determined whether the SDR option has been selected, and if so, the sub-process proceeds to step 1320 to execute an SDR program. Alternatively, if it is determined at 1318 that the SDR option has not been selected, the process proceeds to "C3" (to FIG. 33D).

FIG. 33D shows a flow diagram illustrating a sub-process at 1330 for defining a duration for a schedule multicasting existing assets. The sub-process 1330 proceeds from "C3" (from FIG. 33C) to step 1332 in which an interface, allowing a user to select playback duration program options, is provided. The playback duration program options include a run option, a repeat option, and a playback interval option. Also in step 1332, user input indicative of a selected duration option is received. The sub-process 1330 is identical to the sub-process 930 (FIG. 23C).

FIG. 33E shows a flow diagram illustrating a sub-process at 1350 illustrating further steps of the sub-process 1250 (FIG. 33A) for defining and scheduling a multicast operation. The depicted process proceeds from "C12" (from FIG. 33D) to step 1352 in which a set of multi-casting commands and associated parameters is generated for each defined multi-casting action. Each set of multi-casting commands and parameters is generated based on: multi-casting schedule information including the multi-casting start date and start time defined for the associated

multi-casting action, and the multi-casting duration program defined for the associated multi-casting action; and the associated multi-casting destinations defined for the associated multi-casting action. From step 1352, the sub-process proceeds to step 1353 in which the commands and associated parameters associated with each of the multi-casting actions are transmitted from the administrator terminal 12 (FIG. 1) to the corresponding selected multi-casting media server for executing a corresponding multi-casting operation in accordance with the multi-casting schedule information.

For example, in the preferred embodiment, a set of multi-casting commands and parameters is transmitted from the administrator terminal to a selected media server. The set of multi-casting commands and parameters programs the selected media server to execute multi-casting operations beginning on the play-back start date and start time specified using the start box 1236 (FIG. 32), in accordance with a multi-casting schedule specified by the user via the GUI components of the duration box 1260 (FIG. 32). Each iteration of the multi-casting operation includes streaming the asset to the multi-casting destination locations specified by the user via the destination specification box 1240 (FIG. 32) of the multi-casting GUI screen.

From step 1353, the sub-process proceeds to 1354 at which it is again determined whether the notification option has been selected, and if not, the process proceeds to "C" (to FIG. 27). In the preferred embodiment, the determination at 1354 is implemented using the enable notification check box 1244 (FIG. 32). Alternatively, if it is determined at 1354 that the notification option has been selected, the sub-process proceeds to step 1355 in which a set of notification commands and associated notification parameters is generated for each defined notification action. Each set of notification commands and notification parameters is generated based on user input received in response to GUI screens for defining selected circumstances under which selected notification messages are to be provided to selected end users. From step 1355, the sub-process proceeds to step 1356 in which the notification commands and associated notification parameters associated with

each of the play-back actions are transmitted from the administrator terminal 12 (FIG. 1) to the corresponding selected media server for executing a corresponding notification operation.

FIG. 34 shows a block diagram of a selectable calendar GUI screen at 1360, the screen 1360 including forward and reverse selection buttons 1362 and 1364 for displaying days and dates of successive months in a calendar view. The user may pick a date by selecting a number representing a date as indicated at 1366. The screen 1360 is displayed and activated upon activation of the start date drop down list 1236 (FIG. 32) of the multicasting GUI screen.

FIG. 35 shows a block diagram illustrating a global schedule summary Web document at 1400 in accordance with the present invention, the Web document 1400 including graphical information indicative of a plurality of events, or operations, which have been scheduled for execution at corresponding media servers by a particular user, or administrator, using the media asset management and scheduling GUI process of the present invention as described above. The Web document 1400 is provided by a global asset management server which receives the server identification information, schedule information, and operation information associated with each media operation scheduled by authorized users, or administrators. In order to provide the Web document 1400, the asset management and scheduling GUI process of the present invention includes further steps for transmitting the server identification information, schedule information, and operation information associated with each user defined media operation from the administrator terminal 12 (FIG. 1) to the global asset management server. The global asset management server stores a global schedule summary file for each authorized user, the summary file including all server identification information, operation information, and scheduling information associated with each operation, or event, scheduled by the corresponding user via the asset management and scheduling GUI process of the present invention described above.

In one embodiment of the present invention, the Web document 1400 is provided by SERVER_1 (FIG. 1) which executes the asset management and scheduling program 23 (FIG. 1), and which serves as the global asset management server. In this embodiment, SERVER_1 receives the server identification information, schedule information, and operation information associated with each media operation from the administrator terminal 12 (FIG. 1) at the same time that the commands and associated parameters associated with the media operations are transmitted to the media server indicated by the server identification information. The global asset management server provides a centralized location for storage of information associated with all scheduled media operations. The Web document 1400 may be accessed by a user via a Web browser application executed by the administrator terminal 12 (FIG. 1) using a corresponding URL. In a preferred embodiment of the present invention, the Web document 1400 is a HTML document including embedded Java applets which are executed upon activation of corresponding active areas of the HTML document as further explained below.

The Web document 1400 includes: a schedule time interval selection area 1402 providing a group of GUI components which allow the user to select a time interval, a time zone, a date, and a start time for defining a selected time interval of the global schedule summary; and a global schedule display area 1404 for displaying event programming information indicative of media operations scheduled for execution during the selected time interval defined using the GUI components of the time interval selection area 1402 as further explained below.

The time interval selection area 1402 includes: a time interval drop down list 1406 allowing the user to select a time interval; a time zone drop down list 1410 allowing the user to select from a plurality of time zones, the selected time zone providing a reference for the selected time interval; a date drop down list 1412 allowing the user to select a date for the selected time interval of the global schedule summary associated with the particular user; and a start time drop down list 1414

allowing the user to select a start time for defining the selected time interval of the global schedule summary associated with the particular user.

The program schedule area 1404 includes: a plurality of entries 1418 for displaying graphical indicia indicative of information associated with corresponding media operations which have been defined and schedule by the particular user for execution by a selected media server during the selected time interval; a plurality of time interval columns 1420 delineating time sub-intervals of the selected time interval of the global event schedule summary; and a subject column 1422 for displaying graphical subject indicia indicative of a subject associated with at least one corresponding media operation which has been defined and scheduled by the particular user to be executed by a selected media server during the selected time interval.

For each of the entries 1418, the program schedule area includes corresponding subject indicia 1424 indicative of a subject associated with at least one corresponding media operation, and corresponding operation indicia 1426 indicative of corresponding media operations to be performed by corresponding selected media servers during a corresponding time sub-interval indicated by the particular time interval columns 1420 between which the operation indicia 1426 is located. Each operation indicia 1426 is associated with an active area as further explained below.

For example, an entry of the subject column 1422 may include a subject indicia 1424 indicative of a particular course offered by an educational institution (e.g., Engineering 101), and an operation indicia 1426 indicative of a selected media operation to be performed on an asset comprising audio and video data representing a lecture associated with the particular course. The selected media operation indicated by the operation indicia 1426 may be selected from the above defined media operations including copy operations, delete operations, encoding operations, multicasting operations, and notification operations.

Upon selection of one of the active areas associated with one of the operation indicia 1426, an applet is activated, and interpreted by a Web browser at the administrative terminal 12 (FIG. 1), the applet providing at least a portion of the asset management and scheduling GUI process of the present invention. Clicking on, or activating, one of the active areas associated with one of the operation indicia 1426 causes the GUI process of the present invention to be executed, and initialized to display a corresponding one of the GUI screens described above which is associated with defining and scheduling the media operation associated with the selected operation indicia 1426. The user may then use the GUI process of the present invention to redefine, reschedule, or cancel the corresponding media operation associated with the selected operation indicia 1426 as further explained below.

FIG. 36 shows a flow diagram illustrating a process at 1450 for managing global schedule information in accordance with the present invention. In one embodiment, the process 1450 is implemented as computer readable code executed by the processing unit of SERVER_1 (FIG. 1) which provides a global asset management server. The process 1450 begins with step 1452 in which summary information associated with a particular media operation is received from the administrator terminal 12 (FIG. 1). The summary information includes: user identification information indicating a corresponding user who has defined and scheduled the particular media operation using the asset management and scheduling GUI process described above; server identification information indicating a selected media server scheduled to execute the particular media operation; operation information defining a particular media operation (e.g., defining the particular media operation as a copy asset operation, a delete asset operation, an encoding asset operation, a multicasting operation, or a notification operation); and schedule information specifying a schedule for executing the particular media operation. In an embodiment, the summary information associated with each particular media operation is transmitted from the administrator terminal 12 to SERVER_1 (FIG. 1) at the same time that the commands and associated parameters associated with the particular

media operation are transmitted from the administrator terminal to the selected media server which is indicated by the server identification information.

From step 1452, the process proceeds to step 1454 in which the server identification information, operation information, and schedule information associated with the particular media operation is stored in a global schedule file associated with the indicated user. In an embodiment, the global schedule file associated with the indicated user is stored on the local disk 22 (FIG. 1) of SERVER_1.

FIG. 37 shows a flow diagram illustrating a process at 1470 for generating a global schedule Web document in accordance with the present invention. In the described embodiment, the process 1470 is implemented as computer readable code executed by the processing unit of SERVER_1 (FIG. 1) which acts as the global asset management server. The process 1470 begins with step 1472 in which user information specifying a user, or administrator, is received at SERVER_1 from the administrator terminal 12 (FIG. 1) via the network. From step 1472, the process proceeds to step 1474 in which the global asset management server reads a global schedule file associated with the user specified in step 1472. From step 1474, the process proceeds to step 1476 in which the global asset management server generates a global schedule Web document associated with the specified user based on the global schedule file associated with the user. The global schedule Web document includes: a plurality of operation graphical indicia associated with corresponding operations defined by the specified user; server identification information, operation information, and schedule information associated with each corresponding operation graphical indicia; active areas associated with each corresponding operation graphical indicia; an applet providing the asset management and scheduling GUI process of the present invention; and links to media servers indicated by the server identification information. In one embodiment, the global schedule Web document 1400 (FIG. 35) provides the format for the document generated in step 1476.

FIG. 38 shows a flow diagram illustrating a process at 1500 for reviewing a summary of all scheduled media operations associated with a particular user of the asset management and scheduling GUI process of the present invention, the process 1500 also providing for editing operation information and schedule information associated with selected ones of the scheduled media operations associated with the particular user. In the described embodiment, the process 1500 is implemented as computer readable code executed by the processing unit of the administrator terminal 12 (FIG. 1). The process 1500 begins with step 1501 in which user identification information specifying a user is transmitted from the administrator terminal 12 (FIG. 1) to the global asset management server, that is SERVER_1 (FIG. 1) in the described embodiment. In one embodiment, step 1501 includes accessing SERVER_1 via the Web browser using a predetermined uniform resource locator (URL). From step 1501, the process proceeds to step 1502 in which a global schedule Web document associated with the specified user is received at the administrator terminal 12 (FIG. 1). The global schedule Web document includes: a plurality of operation graphical indicia associated with corresponding operations defined by the specified user; server identification information, operation information, and schedule information associated with each corresponding operation graphical indicia; active areas associated with each corresponding operation graphical indicia; an applet providing the asset management and scheduling GUI process of the present invention; and links to media servers indicated by the server identification information. In the preferred embodiment, the Web document is implemented by the Web document 1400 (FIG. 35).

In step 1504, the Web document is displayed in a browser window of the administrative terminal 12 (FIG. 1). From step 1504, the process proceeds to step 1506 in which user input indicating a selected operation graphical indicia, associated with a selected media operation, is received. In the preferred embodiment, the user selects an operation graphical indicia 1426 (FIG. 35) by selecting (e.g., "clicking on") an active area which is coincident with the selected operation graphical indicia (FIG. 35). In step 1507, an applet embedded in the Web document is executed at

the administrator terminal to provide the asset management and scheduling GUI process of the present invention as described above. In the preferred embodiment, the applet is a Java applet which is embedded in the Web document 1400 (FIG. 35).

In step 1508, operation information and schedule information associated with a selected scheduled event is read. In step 1510, the asset management and scheduling GUI process of the present invention is initiated to display specific GUI screens for editing the operation information and schedule information associated with the selected scheduled event. In step 1510, the asset management and scheduling GUI process advances to the GUI screen associated with the scheduled event. For example, if the scheduled event is a copy action or a delete action, the asset management and scheduling GUI process proceeds to "A" (to FIG. 6). As another example, if the selected scheduled event is a scheduled encoding operation, the process proceeds to "B" (to FIG. 13A). As yet a further example, if the selected scheduled event is a scheduled multicasting operation, the process proceeds to "C" (back to FIG. 33A). In any of these cases, the GUI process displays the appropriate GUI screen having the operation information and schedule information associated with the selected media operation displayed therein for editing.

From step 1510, the process proceeds to step 1512 in which a link is established with the media server indicated by the server identification information associated with the selected media operation. In step 1514, existing commands and associated parameters associated with the selected event are deleted at the source media server associated with the selected event. From step 1514, the process proceeds to "J" (to FIG. 4) to execute the asset management and scheduling GUI process in order to redefine, reschedule, or cancel the selected media operation.

Although the present invention has been particularly shown and described above with reference to a specific embodiment, it is anticipated that alterations and modifications thereof will

no doubt become apparent to those skilled in the art. It is therefore intended that the following claims be interpreted as covering all such alterations and modifications as fall within the true spirit and scope of the invention.

The embodiments of the invention in which an exclusive property or privilege is claimed are defined as follows:

1. A method for providing a centralized user interface at an administrator terminal in a network for managing operations including encoding operations performed on media data by selected ones of a plurality of media servers, the process comprising the steps of:

displaying a graphical user interface at the administrator terminal, said interface including a plurality of interface components enabling a user to select between a manual encoding mode for manually starting and stopping manual encoding processes to be performed by selected ones of the servers, and a scheduled encoding mode for defining schedules for scheduled encoding processes to be performed by selected ones of the servers;

receiving information input by the user specifying a selected one of the media servers, a selected mode of operation, and an encoding operation;

generating commands and associated parameters based on the input information for instructing the selected server to execute the specified encoding operation in accordance with the selected mode of operation; and

transmitting said commands and said associated parameters to said selected server.

2. A method as recited in claim 1 wherein each of the media servers is communicatively coupled with at least one corresponding multimedia device capable of generating media data, and wherein each of the media servers is operative to activate a selected one of said multimedia devices, and also operative to encode the media data generated by said selected multimedia device.

3. A method as recited in claim 2 wherein the encoding operations include a preview operation, and wherein the administrator terminal also includes a processing unit, a browser application executed by the processing unit, and a display unit, and wherein the selected multimedia device is

video camera which generates media data including video data, and wherein said steps of displaying, receiving, generating, and transmitting further comprise:

displaying a preview select interface component enabling the user to select a preview option for executing a preview operation including displaying video data generated by a selected video camera coupled with said selected server;

receiving information input by the user indicating that said preview option has been selected;

generating preview commands and associated preview parameters for instructing said selected server to execute said preview operation;

transmitting said preview commands and said associated preview parameters to said selected server;

receiving preview video data from said selected server; and

displaying said preview video data in a browser window on the display unit of the administrator terminal

4. A method as recited in claim 1 wherein the network is an internet protocol (IP) network.

5. A method as recited in claim 1 wherein the administrator terminal also includes a processing unit, a browser application executed by the processing unit, and a display unit, and wherein said process is initiated by performing the steps of:

transmitting an applet to the administrator terminal via the network; and

executing said applet over the processing unit of the administrator terminal, said graphical user interface being displayed within a browser window generated by said browser application on the display unit.

6. A process as recited in claim 2 wherein if said scheduled encoding mode is selected, said steps of displaying a graphical user interface, receiving information, and generating commands

further comprise the steps of:

displaying scheduling interface components enabling the user to select a start time and a start date for the scheduled encoding operation;

receiving information input by the user indicating a selected start time and a selected start date for initiating said scheduled encoding operation; and

generating encoding commands and associated encoding parameters for instructing said selected server to encode media data received from said selected multimedia device at said selected start time on said selected start date.

7. A method as recited in claim 6 wherein said steps of displaying said scheduling interface components, and receiving information further include the steps of:

displaying duration interface components enabling the user to select from time duration specification options including, selecting a scheduled stop date and stop time for terminating said encoding operation, and

selecting a time duration for which said encoding operation is to continue following said selected start time on said selected start date; and

receiving information input by the user indicative of a selected time duration specification option.

8. A method as recited in claim 1 wherein each of the servers is operative to access at least one corresponding memory device for storing media data, and wherein each of the servers is further operative to record selected portions of encoded media data that are encoded during said encoding operations, and wherein said steps of displaying a graphical user interface, receiving information, and generating commands and parameters further comprise the steps of:

displaying a storage selection interface component enabling the user to select a storage location by browsing a list of available storage locations including predetermined mapped ones of

the memory devices associated with said selected server;

receiving information input by the user indicating a selected storage location; and

generating record commands and associated record parameters for instructing said selected server to store the selected portion of encoded media data at said selected storage location.

9. A method as recited in claim 8 wherein said steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying a name selection interface component enabling the user specify an asset name for said encoded portion of media data to be stored at said selected storage location; and

receiving information input by the user indicating a specified asset name;

wherein said record commands and associated record parameters provide for instructing said selected encoder media server to store said encoded portion of media data at said selected storage location under said specified asset name.

10. A method as recited in claim 9 wherein said steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying an overwriting option selection interface component enabling the user to select from a plurality of overwriting options for specifying attributes of the recording operation if a file having said specified asset name exists at said selected storage location at the time of execution of said recording operation, said overwriting options including,

a first option for specifying that the user is to be prompted if a file having said specified asset name exists at said selected storage location at the time of execution of said recording operation,

a second option for specifying that any file having said specified asset name, existing at said selected storage location at the time of execution of said recording operation, is to be overwritten with said encoded portion of media data, and

a third option for specifying that any file having said specified asset name, existing at said selected storage location at the time of execution of said recording operation, is to be appended with said encoded portion of media data.

11. A method as recited in claim 1 wherein the network further includes a plurality of end user terminals communicatively coupled to the administrator terminal and to the servers, each of the servers being further operative to stream media data to selected ones of the end user terminals, the operations further including streaming operations, and wherein said steps of displaying a graphical user interface, receiving information, and generating commands and associated parameters further comprise the steps of:

displaying streaming interface components enabling the user to define a streaming operation for streaming the encoded media data to selected ones of the end user terminals receiving streaming operation information input by the user; and

generating streaming commands and associated streaming parameters based on said streaming operation information, said streaming commands and associated streaming parameters for instructing said selected server to execute a streaming operation for streaming the encoded portion of media data from said selected encoder media server to corresponding selected ones of the end user terminals via the network.

12. A method as recited in claim 11 wherein said steps of displaying streaming interface components, and receiving information further comprise the steps of:

displaying streaming destination selection interface components enabling the user to select at least one of the end user terminals as a destination for streaming said encoded portion of media data; and

receiving information input by the user indicative of at least one selected end user;

wherein said commands and associated parameters further include a streaming command and

associated streaming parameters for instructing said selected encoder media server to stream said encoded portion of media data to said selected end users.

13. A method as recited in claim 11 wherein the network further includes a plurality of end user terminals communicatively coupled to the administrator terminal and to the servers, each of the servers being further operative to stream media data to selected ones of the end user terminals, the operations further including streaming operations, and wherein the operations further include playback operations for streaming the stored portion of encoded media data from said selected server to corresponding selected ones of the end user terminals in accordance with a user defined playback schedule.

14. A method as recited in claim 13 wherein said steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying playback destination selection interface components enabling the user to select at least one of the end user terminals as a destination for streaming said encoded portion of media data in accordance with a user defined playback schedule;

displaying play-back schedule interface components enabling the user define a play-back schedule; and

receiving information input by the user indicative of a user defined play-back schedule;

wherein said commands and associated parameters further include a play-back command and associated play-back parameters for instructing said selected encoder media server to stream said stored portion of encoded media data to said selected end users via the network in accordance with said user defined play-back schedule.

15. A method as recited in claim 14 wherein said play-back schedule interface components comprise:

a first group of components enabling the user to select a start time and a start date for said play-back schedule; and

a second group of components enabling the user to select from a plurality of options for specifying a play-back schedule duration.

16. A method as recited in claim 13 wherein said operations further include notification operations associated with corresponding ones of the playback operations, said notification operations for sending notification messages to selected network addresses associated with selected ones of the end user terminals and the administrator terminal.

17. A method as recited in claim 16 wherein the notification operations include determining success or failure of the streaming specified by the corresponding playback operation, and wherein said steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying notification circumstance selection interface components enabling the user to select from the notification circumstance options of,

a first option of sending a notification message to selected network addresses if the streaming specified by the corresponding playback operation fails,

a second option of sending a notification message to selected network addresses if the streaming specified by the corresponding playback operation is successful, and

a third option of sending a notification message to selected network addresses if the streaming specified by the corresponding playback operation fails of is successful; and

receiving information input by the user indicating a selected notification circumstance option;

wherein said commands and associated parameters further include a notification command

and associated notification parameters for instructing said selected encoder media server to send notification messages to the selected network addresses in accordance with said selected notification circumstance option.

18. A method as recited in claim 2 wherein the administrator terminal also includes a processing unit, a browser application executed by the processing unit, and a display unit, and wherein a selected one of the multimedia devices is a video camera which generates media data including video data, and wherein if said manual encoding mode is selected, said process further comprises the steps of:

generating manual mode initiation commands and associated parameters for instructing the selected server to initiate a manual encoding operation;

transmitting said manual mode initiation commands and associated parameters to said selected server via the network;

receiving video data from said selected server via the network;

displaying said video data in a browser window on the display unit of the administrator terminal, and also displaying manual encoding mode control interface components enabling the user to select from manual control options including a start option for starting and resuming the manual encoding process, and a stop option for stopping the manual encoding process;

receiving information input by the user indicating a selected one of the manual control options;

if said start option is selected, generating a start command for instructing said selected media server to start the manual encoding process, and transmitting said start command to said selected server; and

if said stop option is selected, generating a stop command for instructing said selected media server to stop the manual encoding process, and transmitting said stop command to said selected server.

19. A method for providing a centralized user interface at an administrator terminal in a network for remotely controlling encoding operations performed on media data by selected ones of a plurality of media servers, the process comprising the steps of:

generating initiation commands and associated initiation parameters for instructing a selected server to initiate a manual encoding operation;

transmitting said initiation commands and said associated initiation parameters to said selected server via the network;

receiving encoded media data from said selected server via the network;

displaying said encoded media data at the administrator terminal, and also displaying interface components enabling the user to select from manual control options including a start option for starting and resuming the manual encoding process, and a stop option for stopping the manual encoding process;

receiving information input by the user indicating a selected one of the manual control options;

if said start option is selected, generating a start command for instructing said selected media server to start the manual encoding process, and transmitting said start command to said selected server; and

if said stop option is selected, generating a stop command for instructing said selected media server to stop the manual encoding process, and transmitting said stop command to said selected server.

20. A method as recited in claim 19 wherein each of the media servers is communicatively coupled with at least one corresponding multimedia device capable of generating media data, and wherein each of the media servers is operative to activate a selected one of said multimedia devices, and also operative to encode the media data generated by said selected multimedia device, and wherein said interface components include a multimedia device enabling a user to select from said

multimedia devices.

21. A method as recited in claim 19 wherein the administrator terminal includes a processing unit, and a viewing application executed by the processing unit, and wherein said step of displaying said encoded media data at the administrator terminal includes displaying said encoded media data in a window generated by said viewing application.

22. A machine readable storage device having stored therein encoding instructions for executing a process of providing a centralized user interface at an administrator terminal in a network for managing operations including encoding operations performed on media data by selected ones of a plurality of media servers, the process comprising the steps of:

displaying a graphical user interface at the administrator terminal, said interface including a plurality of interface components enabling a user to select between a manual encoding mode for manually starting and stopping manual encoding processes to be performed by selected ones of the servers, and a scheduled encoding mode for defining schedules for scheduled encoding processes to be performed by selected ones of the servers;

receiving information input by the user specifying a selected one of the media servers, a selected mode of operation, and an encoding operation;

generating commands and associated parameters based on the input information for instructing the selected server to execute the specified encoding operation in accordance with the selected mode of operation; and

transmitting said commands and said associated parameters to said selected server.

23. A machine readable storage device as recited in claim 22 wherein each of the media servers is communicatively coupled with at least one corresponding multimedia device capable of generating

media data, and wherein each of the media servers is operative to activate a selected one of said multimedia devices, and also operative to encode the media data generated by said selected multimedia device.

24. A machine readable storage device as recited in claim 23 wherein the encoding operations include a preview operation, and wherein the administrator terminal also includes a processing unit, a browser application executed by the processing unit, and a display unit, and wherein the selected multimedia device is a video camera which generates media data including video data, and wherein said steps of displaying, receiving, generating, and transmitting further comprise:

displaying a preview select interface component enabling the user to select a preview option for executing a preview operation including displaying video data generated by a selected video camera coupled with said selected server;

receiving information input by the user indicating that said preview option has been selected;

generating preview commands and associated preview parameters for instructing said selected server to execute said preview operation;

transmitting said preview commands and said associated preview parameters to said selected server;

receiving preview video data from said selected server; and

displaying said preview video data in a browser window on the display unit of the administrator terminal.

25. A machine readable storage device as recited in claim 22 wherein the network is an internet protocol (IP) network.

26. A machine readable storage device as recited in claim 22 wherein the administrator terminal also includes a processing unit, a browser application executed by the processing unit, and a display

unit, and wherein said process is initiated by performing the steps of:

transmitting an applet to the administrator terminal via the network; and

executing said applet over the processing unit of the administrator terminal, said graphical user interface being displayed within a browser window generated by said browser application on the display unit.

27. A machine readable storage device as recited in claim 23 wherein if said scheduled encoding mode is selected, said steps of displaying a graphical user interface, receiving information, and generating commands further comprise the steps of:

displaying scheduling interface components enabling the user to select a start time and a start date for the scheduled encoding operation;

receiving information input by the user indicating a selected start time and a selected start date for initiating said scheduled encoding operation; and

generating encoding commands and associated encoding parameters for instructing said selected server to encode media data received from said selected multimedia device at said selected start time on said selected start date.

28. A machine readable storage device as recited in claim 27 wherein said steps of displaying said scheduling interface components, and receiving information further include the steps of:

displaying duration interface components enabling the user to select from time duration specification options including,

selecting a scheduled stop date and stop time for terminating said encoding operation, and

selecting a time duration for which said encoding operation is to continue following said selected start time on said selected start date; and

receiving information input by the user indicative of a selected time duration specification

option.

29. A machine readable storage device as recited in claim 22 wherein each of the servers is operative to access at least one corresponding memory device for storing media data, and wherein each of the servers is further operative to record selected portions of encoded media data that are encoded during said encoding operations, and wherein said steps of displaying a graphical user interface, receiving information, and generating commands and parameters further comprise the steps of:

displaying a storage selection interface component enabling the user to select a storage location by browsing a list of available storage locations including predetermined mapped ones of the memory devices associated with said selected server;

receiving information input by the user indicating a selected storage location; and

generating record commands and associated record parameters for instructing said selected server to store the selected portion of encoded media data at said selected storage location.

30. A machine readable storage device as recited in claim 29 wherein said steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying a name selection interface component enabling the user specify an asset name for said encoded portion of media data to be stored at said selected storage location; and

receiving information input by the user indicating a specified asset name;

wherein said record commands and associated record parameters provide for instructing said selected encoder media server to store said encoded portion of media data at said selected storage location under said specified asset name.

31. A machine readable storage device as recited in claim 30 wherein said steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying an overwriting option selection interface component enabling the user to select from a plurality of overwriting options for specifying attributes of the recording operation if a file having said specified asset name exists at said selected storage location at the time of execution of said recording operation, said overwriting options including,

a first option for specifying that the user is to be prompted if a file having said specified asset name exists at said selected storage location at the time of execution of said recording operation,

a second option for specifying that any file having said specified asset name, existing at said selected storage location at the time of execution of said recording operation, is to be overwritten with said encoded portion of media data, and

a third option for specifying that any file having said specified asset name, existing at said selected storage location at the time of execution of said recording operation, is to be appended with said encoded portion of media data.

32. A machine readable storage device as recited in claim 22 wherein the network further includes a plurality of end user terminals communicatively coupled to the administrator terminal and to the servers, each of the servers being further operative to stream media data to selected ones of the end user terminals, the operations further including streaming operations, and wherein said steps of displaying a graphical user interface, receiving information, and generating commands and associated parameters further comprise the steps of:

displaying streaming interface components enabling the user to define a streaming operation for streaming the encoded media data to selected ones of the end user terminals

receiving streaming operation information input by the user; and

generating streaming commands and associated streaming parameters based on said streaming operation information, said streaming commands and associated streaming parameters for instructing said selected server to execute a streaming operation for streaming the encoded portion

of media data from said selected encoder media server to corresponding selected ones of the end user terminals via the network.

33. A machine readable storage device as recited in claim 32 wherein said steps of displaying streaming interface components, and receiving information further comprise the steps of:

displaying streaming destination selection interface components enabling the user to select at least one of the end user terminals as a destination for streaming said encoded portion of media data; and

receiving information input by the user indicative of at least one selected end user;

wherein said commands and associated parameters further include a streaming command and associated streaming parameters for instructing said selected encoder media server to stream said encoded portion of media data to said selected end users.

34. A machine readable storage device as recited in claim 29 wherein the network further includes a plurality of end user terminals communicatively coupled to the administrator terminal and to the servers, each of the servers being further operative to stream media data to selected ones of the end user terminals, the operations further including streaming operations, and wherein the operations further include playback operations for streaming the stored portion of encoded media data from said selected media server to corresponding selected ones of the end user terminals in accordance with a user defined playback schedule.

35. A server operative to provide an applet to a client via a network, the applet including encoding instructions for executing a process of providing a centralized user interface at the client in a network for managing operations including encoding operations performed on media data by selected ones of a plurality of media servers, the process comprising the steps of:

displaying a graphical user interface at the client, said interface including a plurality of

interface components enabling a user to select between a manual encoding mode for manually starting and stopping manual encoding processes to be performed by selected ones of the media servers, and a scheduled encoding mode for defining schedules for scheduled encoding processes to be performed by selected ones of the media servers;

receiving information input by the user specifying a selected one of the media servers, a selected mode of operation, and an encoding operation;

generating commands and associated parameters based on the input information for instructing the selected media server to execute the specified encoding operation in accordance with the selected mode of operation; and

transmitting said commands and said associated parameters to said selected media server.

36. A server recited in claim 35 wherein each of the media servers is communicatively coupled with at least one corresponding multimedia device capable of generating media data, and wherein each of the media servers is operative to activate a selected one of said multimedia devices, and also operative to encode the media data generated by said selected multimedia device.

37. A server recited in claim 36 wherein the encoding operations include a preview operation, and wherein the client also includes a processing unit, a browser application executed by the processing unit, and a display unit, and wherein the selected multimedia device is a video camera which generates media data including video data, and wherein said steps of displaying, receiving, generating, and transmitting further comprise:

displaying a preview select interface component enabling the user to select a preview option for executing a preview operation including displaying video data generated by a selected video camera coupled with said selected media server;

receiving information input by the user indicating that said preview option has been selected;

generating preview commands and associated preview parameters for instructing said

selected media server to execute said preview operation;

transmitting said preview commands and said associated preview parameters to said selected media server;

receiving preview video data from said selected media server; and

displaying said preview video data in a browser window on the display unit of the client.

38. A server recited in claim 35 wherein the network is an internet protocol (IP) network.

39. A server recited in claim 36 wherein if said scheduled encoding mode is selected, said steps of displaying a graphical user interface, receiving information, and generating commands further comprise the steps of:

displaying scheduling interface components enabling the user to select a start time and a start date for the scheduled encoding operation;

receiving information input by the user indicating a selected start time and a selected start date for initiating said scheduled encoding operation; and

generating encoding commands and associated encoding parameters for instructing said selected media server to encode media data received from said selected multimedia device at said selected start time on said selected start date.

40. A server recited in claim 39 wherein said steps of displaying said scheduling interface components, and receiving information further include the steps of:

displaying duration interface components enabling the user to select from time duration specification options including,

selecting a scheduled stop date and stop time for terminating said encoding operation, and

selecting a time duration for which said encoding operation is to continue following

said selected start time on said selected start date; and

receiving information input by the user indicative of a selected time duration specification option.

41. A server recited in claim 35 wherein each of the media servers is operative to access at least one corresponding memory device for storing media data, and wherein each of the media servers is further operative to record selected portions of encoded media data that are encoded during said encoding operations, and wherein said steps of displaying a graphical user interface, receiving information, and generating commands and parameters further comprise the steps of:

displaying a storage selection interface component enabling the user to select a storage location by browsing a list of available storage locations including predetermined mapped ones of the memory devices associated with said selected media server;

receiving information input by the user indicating a selected storage location; and

generating record commands and associated record parameters for instructing said selected media server to store the selected portion of encoded media data at said selected storage location.

42. A server recited in claim 41 wherein said steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying a name selection interface component enabling the user specify an asset name for said encoded portion of media data to be stored at said selected storage location; and

receiving information input by the user indicating a specified asset name;

wherein said record commands and associated record parameters provide for instructing said selected encoder media server to store said encoded portion of media data at said selected storage location under said specified asset name.

43. A server recited in claim 42 wherein said steps of displaying a graphical user interface, and

receiving information further comprise the steps of:

displaying an overwriting option selection interface component enabling the user to select from a plurality of overwriting options for specifying attributes of the recording operation if a file having said specified asset name exists at said selected storage location at the time of execution of said recording operation, said overwriting options including,

a first option for specifying that the user is to be prompted if a file having said specified asset name exists at said selected storage location at the time of execution of said recording operation,

a second option for specifying that any file having said specified asset name, existing at said selected storage location at the time of execution of said recording operation, is to be overwritten with said encoded portion of media data, and

a third option for specifying that any file having said specified asset name, existing at said selected storage location at the time of execution of said recording operation, is to be appended with said encoded portion of media data.

44. A server recited in claim 35 wherein the network further includes a plurality of end user terminals communicatively coupled to the client and to the media servers, each of the media servers being further operative to stream media data to selected ones of the end user terminals, the operations further including streaming operations, and wherein said steps of displaying a graphical user interface, receiving information, and generating commands and associated parameters further comprise the steps of:

displaying streaming interface components enabling the user to define a streaming operation for streaming the encoded media data to selected ones of the end user terminals

receiving streaming operation information input by the user; and

generating streaming commands and associated streaming parameters based on said streaming operation information, said streaming commands and associated streaming parameters for instructing said selected media server to execute a streaming operation for streaming the encoded portion of media data from said selected encoder media server to corresponding selected ones of the end user terminals via the network.

45. A server recited in claim 44 wherein the network further includes a plurality of end user terminals communicatively coupled to the client and to the media servers, each of the media servers being further operative to stream media data to selected ones of the end user terminals, the operations further including streaming operations, and wherein the operations further include playback operations for streaming the stored portion of encoded media data from said selected media server to corresponding selected ones of the end user terminals in accordance with a user defined playback schedule.

46. A server recited in claim 45 wherein said steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying playback destination selection interface components enabling the user to select at least one of the end user terminals as a destination for streaming said encoded portion of media data in accordance with a user defined playback schedule;

displaying play-back schedule interface components enabling the user define a play-back schedule; and

receiving information input by the user indicative of a user defined play-back schedule;

wherein said commands and associated parameters further include a play-back command and associated play-back parameters for instructing said selected encoder media server to stream said stored portion of encoded media data to said selected end users via the network in accordance with said user defined play-back schedule.

47. A server recited in claim 46 wherein said play-back schedule interface components comprise:

a first group of components enabling the user to select a start time and a start date for said play-back schedule; and

a second group of components enabling the user to select from a plurality of options for specifying a play-back schedule duration.

48. A server recited in claim 45 wherein said operations further include notification operations associated with corresponding ones of the playback operations, said notification operations for sending notification messages to selected network addresses associated with selected ones of the end user terminals and the client.

49. A server recited in claim 48 wherein the notification operations include determining success or failure of the streaming specified by the corresponding playback operation, and wherein said steps of displaying a graphical user interface, and receiving information further comprise the steps of:

displaying notification circumstance selection interface components enabling the user to select from the notification circumstance options of,

a first option of sending a notification message to selected network addresses if the

streaming specified by the corresponding playback operation fails,

a second option of sending a notification message to selected network addresses if the streaming specified by the corresponding playback operation is successful, and

a third option of sending a notification message to selected network addresses if the streaming specified by the corresponding playback operation fails of is successful; and

receiving information input by the user indicating a selected notification circumstance option;

wherein said commands and associated parameters further include a notification command and associated notification parameters for instructing said selected encoder media server to send notification messages to the selected network addresses in accordance with said selected notification circumstance option.

50. A server recited in claim 36 wherein the client also includes a processing unit, a browser application executed by the processing unit, and a display unit, and wherein a selected one of the multimedia devices is a video camera which generates media data including video data, and wherein if said manual encoding mode is selected, said process further comprises the steps of:

generating manual mode initiation commands and associated parameters for instructing the selected media server to initiate a manual encoding operation;

transmitting said manual mode initiation commands and associated parameters to said selected media server via the network;

receiving video data from said selected media server via the network;

displaying said video data in a browser window on the display unit of the client, and also

displaying manual encoding mode control interface components enabling the user to select from manual control options including a start option for starting and resuming the manual encoding process, and a stop option for stopping the manual encoding process;

receiving information input by the user indicating a selected one of the manual control options;

if said start option is selected, generating a start command for instructing said selected media server to start the manual encoding process, and transmitting said start command to said selected media server; and

if said stop option is selected, generating a stop command for instructing said selected media server to stop the manual encoding process, and transmitting said stop command to said selected media server.

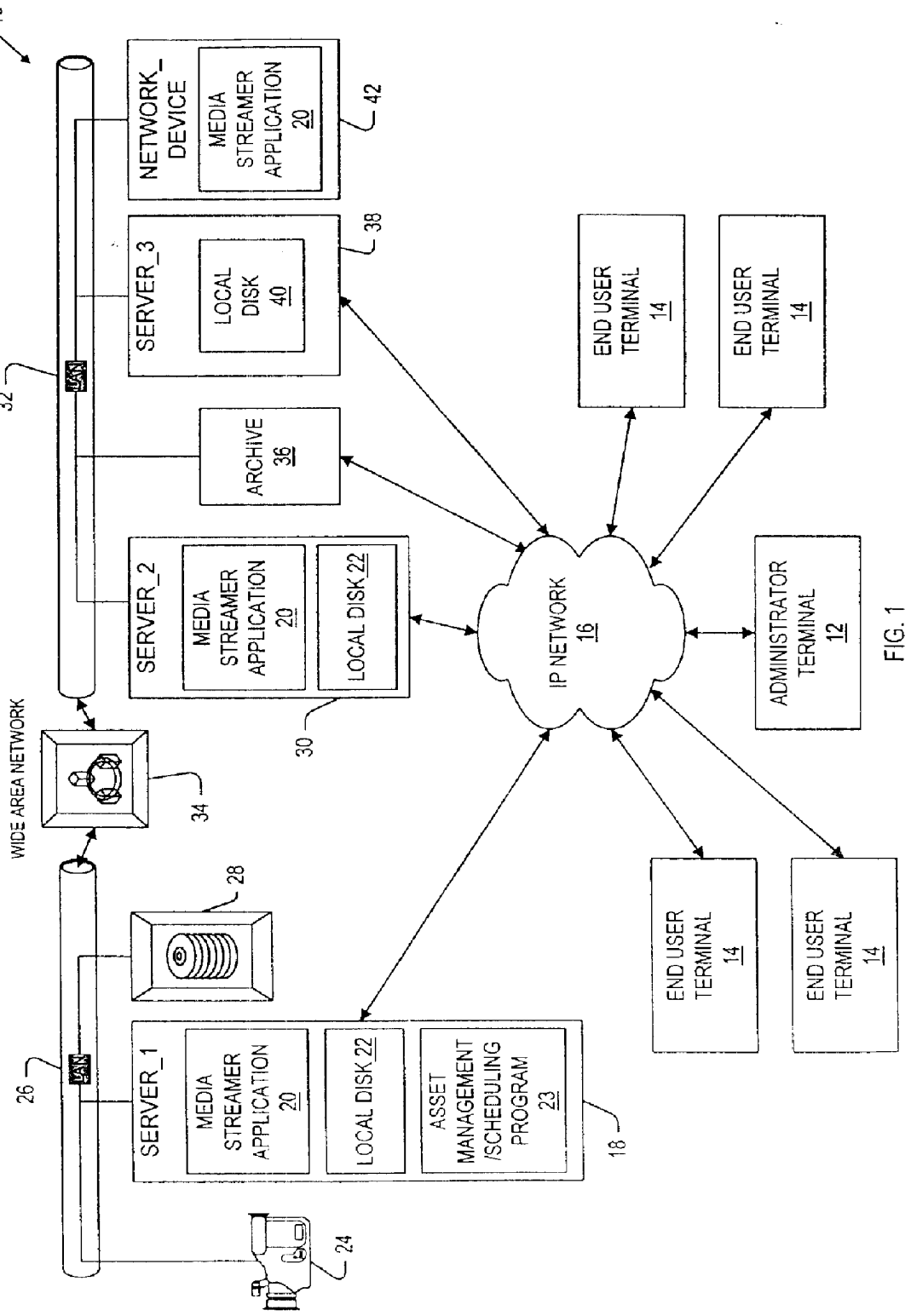

10
NETWORK_ DEVICE
MEDIA STREAMER APPLICATION 20
42
SERVER_3
LOCAL DISK 40
38
ARCHIVE 36
32
SERVER_2
MEDIA STREAMER APPLICATION 20
LOCAL DISK 22
30
WIDE AREA NETWORK
34
28
SERVER_1
MEDIA STREAMER APPLICATION 20
LOCAL DISK 22
ASSET MANAGEMENT /SCHEDULING PROGRAM 23
18
26
24
IP NETWORK 16
END USER TERMINAL 14
END USER TERMINAL 14
END USER TERMINAL 14
END USER TERMINAL 14
ADMINISTRATOR TERMINAL 12

FIG. 1

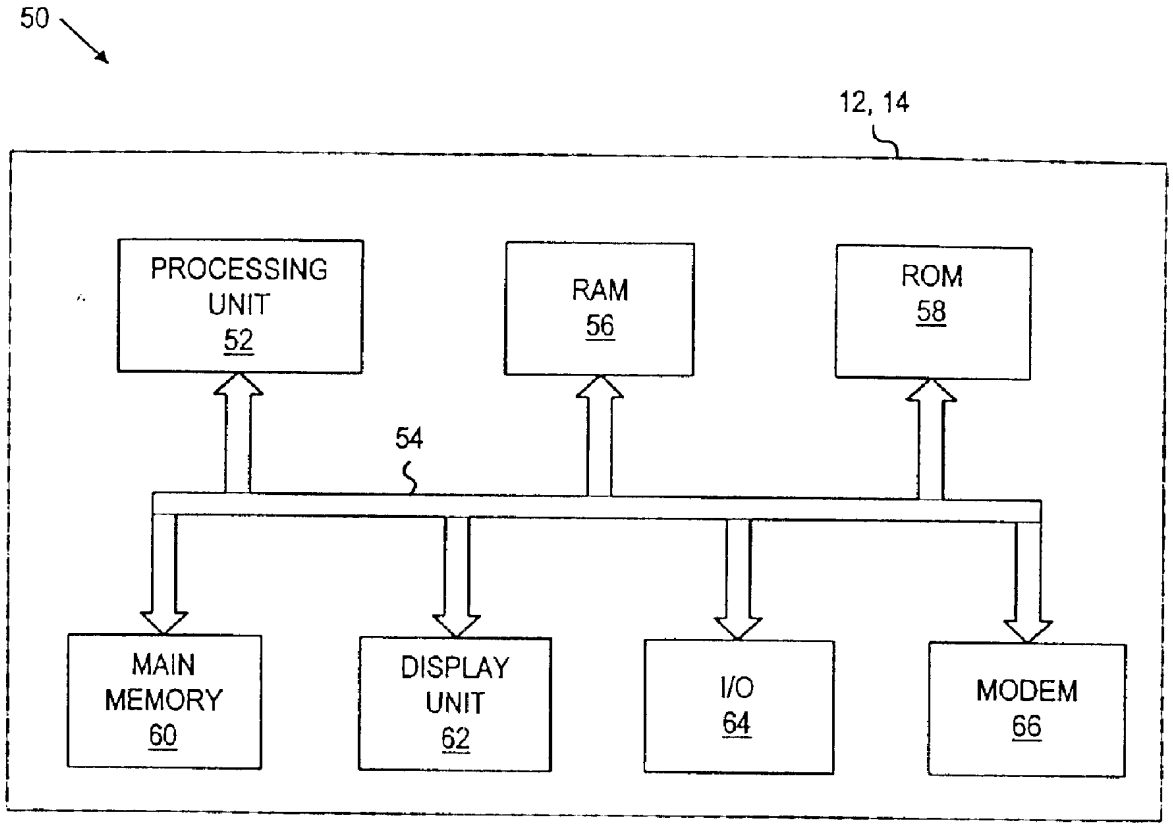

50
12, 14
PROCESSING UNIT 52
RAM 56
ROM 58
54
MAIN MEMORY 60
DISPLAY UNIT 62
I/O 64
MODEM 66

FIG. 2

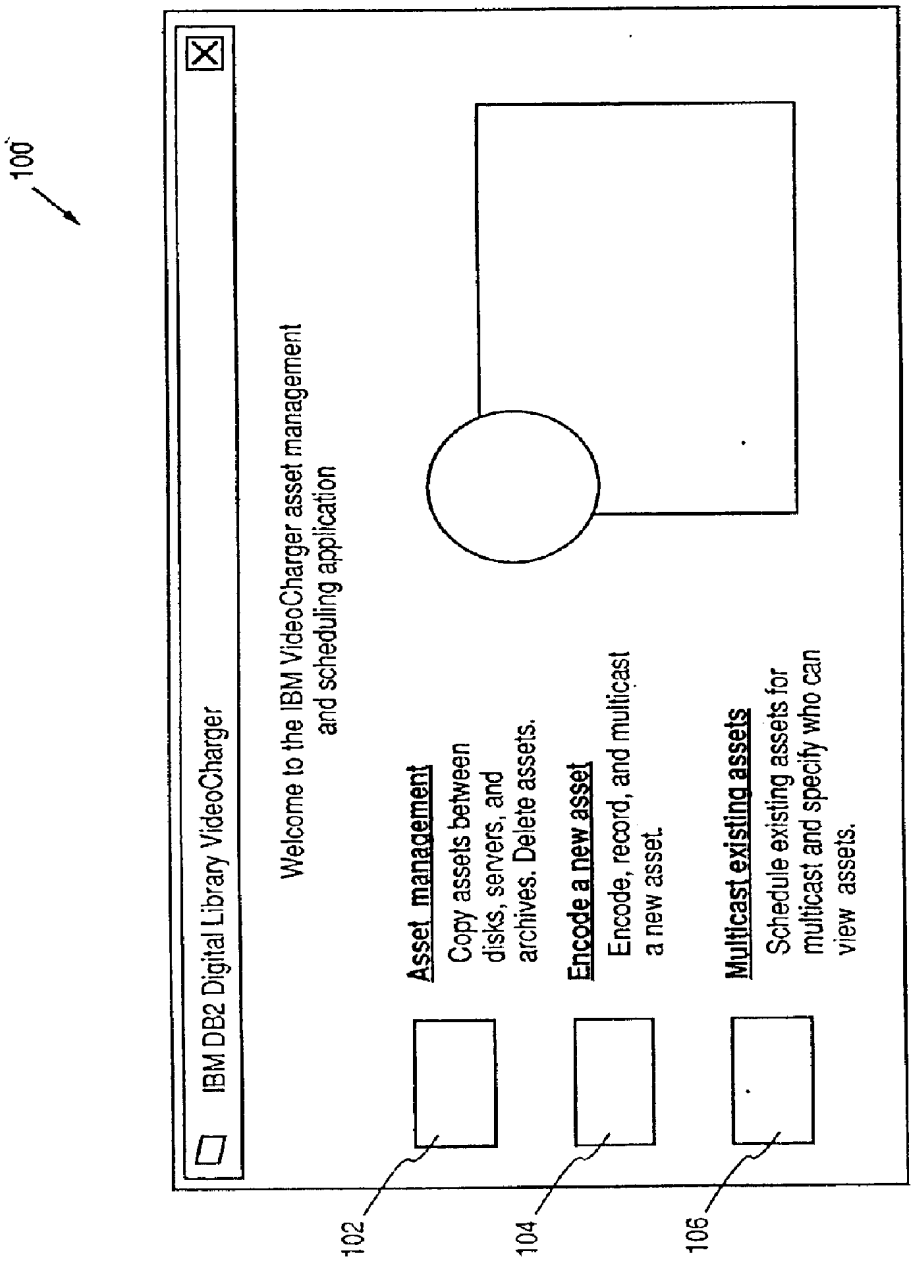

100
IBM DB2 Digital Library VideoCharger
Welcome to the IBM VideoCharger asset management and scheduling application
Asset management
Copy assets between disks, servers, and archives. Delete assets.
Encode a new asset
Encode, record, and multicast a new asset.
Multicast existing assets
Schedule existing assets for multicast and specify who can view assets.
102
104
106

Fig. 3

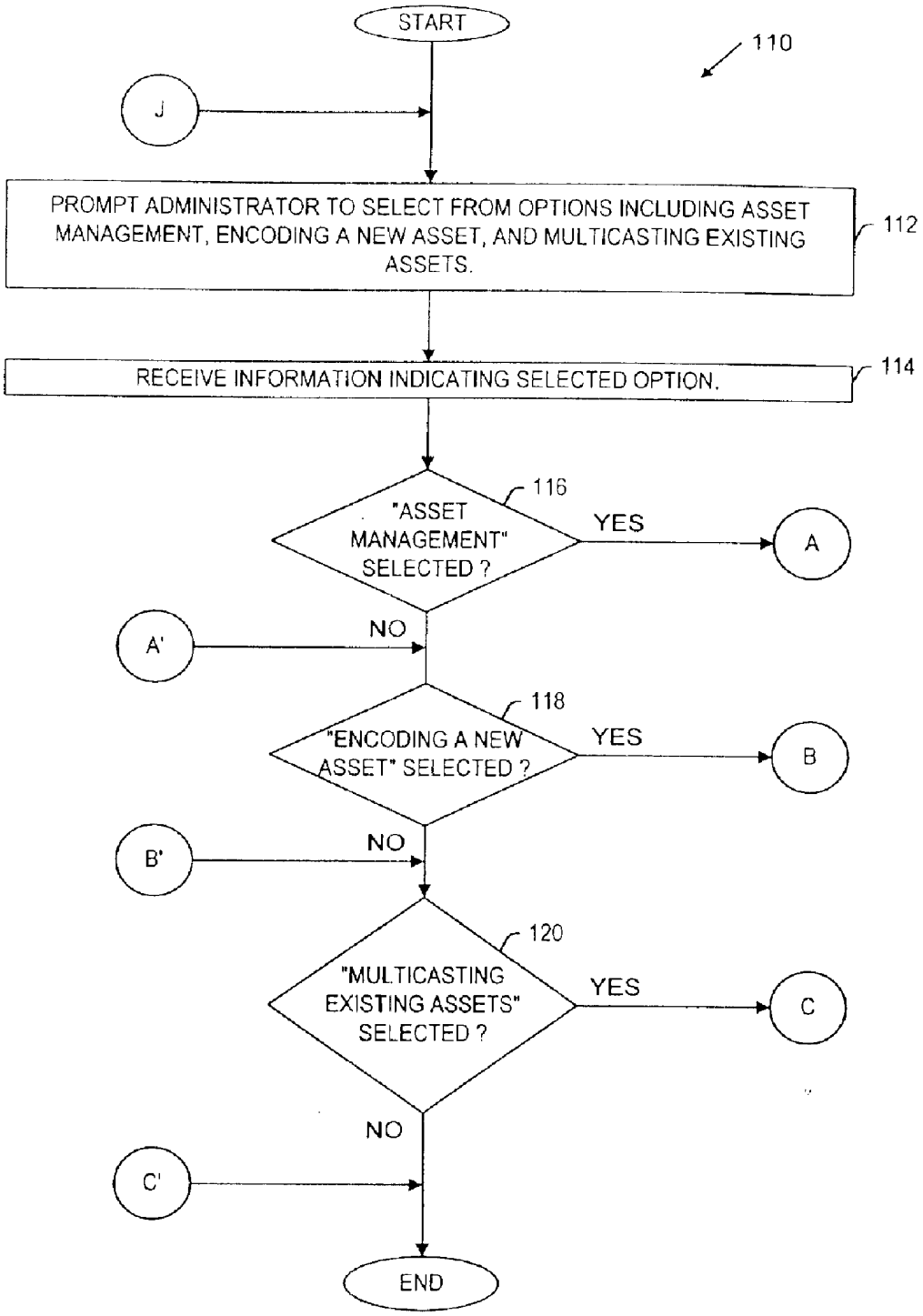

START
110
J
PROMPT ADMINISTRATOR TO SELECT FROM OPTIONS INCLUDING ASSET MANAGEMENT, ENCODING A NEW ASSET, AND MULTICASTING EXISTING ASSETS.
112
RECEIVE INFORMATION INDICATING SELECTED OPTION.
114
"ASSET MANAGEMENT" SELECTED ?
116
YES
A
NO
A'
"ENCODING A NEW ASSET" SELECTED ?
118
YES
B
NO
B'
"MULTICASTING EXISTING ASSETS" SELECTED ?
120
YES
C
NO
C'
END

FIG. 4

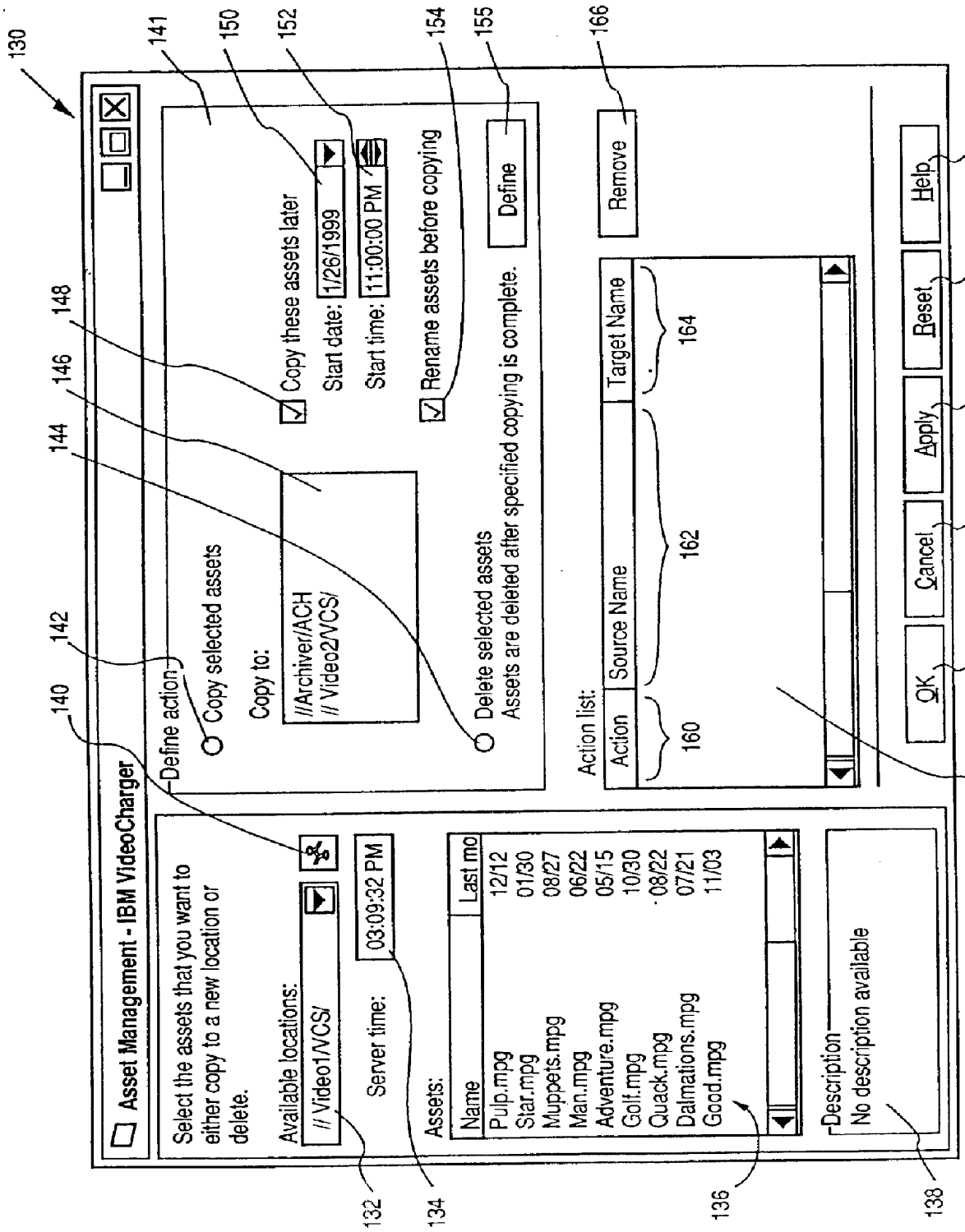

130
141
150
152
154
155
166
148
146
144
142
140
Asset Management - IBM VideoCharger
Select the assets that you want to either copy to a new location or delete.
Available locations:
// Video1/VCS/
Server time:
03:09:32 PM
Assets:
Name
Last mo
Pulp.mpg 12/12
Star.mpg 01/30
Muppets.mpg 08/27
Man.mpg 06/22
Adventure.mpg 05/15
Golf.mpg 10/30
Quack.mpg 08/22
Dalmations.mpg 07/21
Good.mpg 11/03
Description
No description available
Define action
Copy selected assets
Copy to:
//Archiver/ACH
// Video2/VCS/
Copy these assets later
Start date: 1/26/1999
Start time: 11:00:00 PM
Rename assets before copying
Delete selected assets
Assets are deleted after specified copying is complete.
Define
Remove
Action list:
Action
Source Name
Target Name
160
162
164
OK
Cancel
Apply
Reset
Help
168
170
172
174
176
132
134
136
138
156

FIG. 5

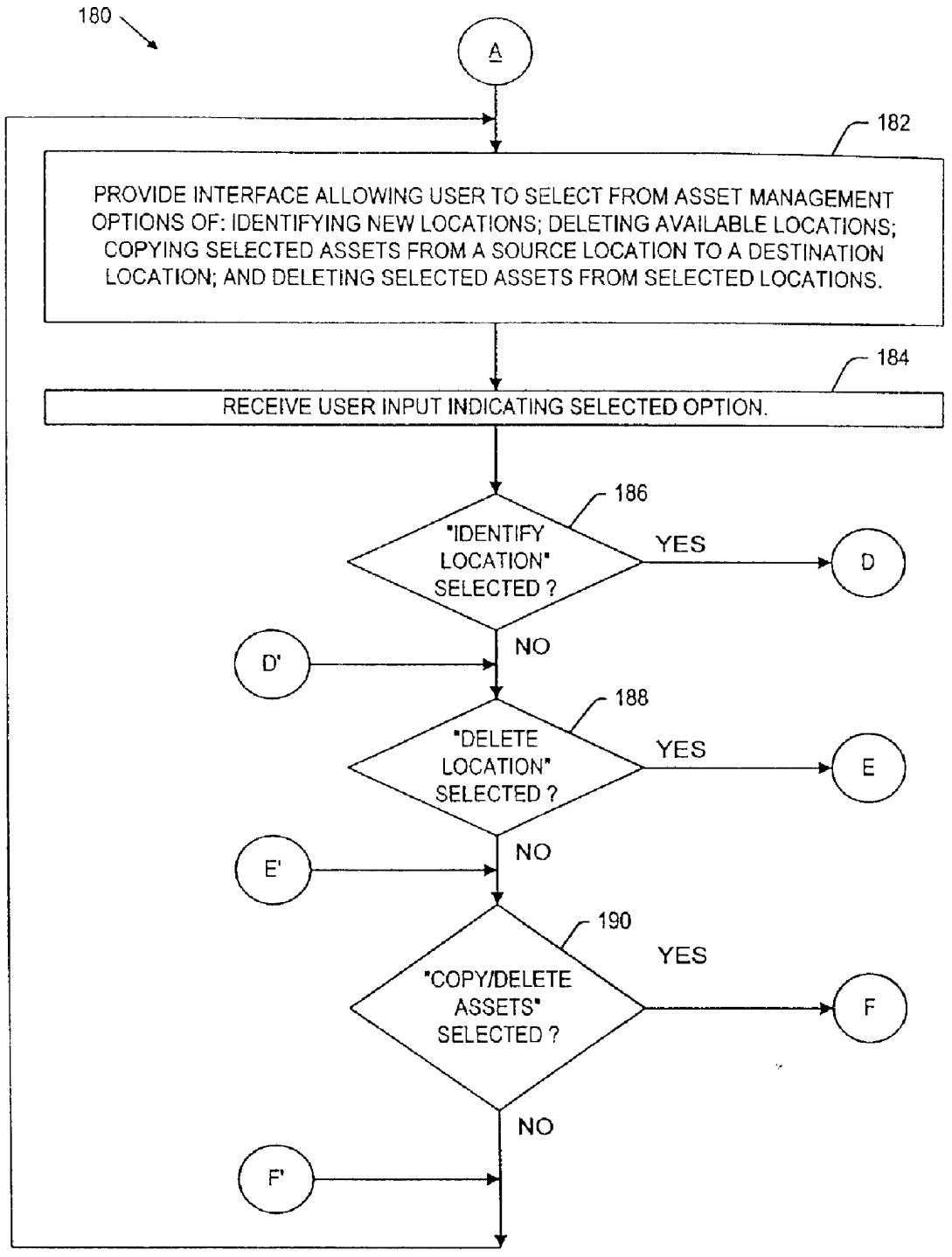

180
A
182
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM ASSET MANAGEMENT OPTIONS OF: IDENTIFYING NEW LOCATIONS; DELETING AVAILABLE LOCATIONS; COPYING SELECTED ASSETS FROM A SOURCE LOCATION TO A DESTINATION LOCATION; AND DELETING SELECTED ASSETS FROM SELECTED LOCATIONS.
184
RECEIVE USER INPUT INDICATING SELECTED OPTION.
186
"IDENTIFY LOCATION" SELECTED ?
YES
D
NO
D'
188
"DELETE LOCATION" SELECTED ?
YES
E
NO
E'
190
"COPY/DELETE ASSETS" SELECTED ?
YES
F
NO
F'

FIG. 6

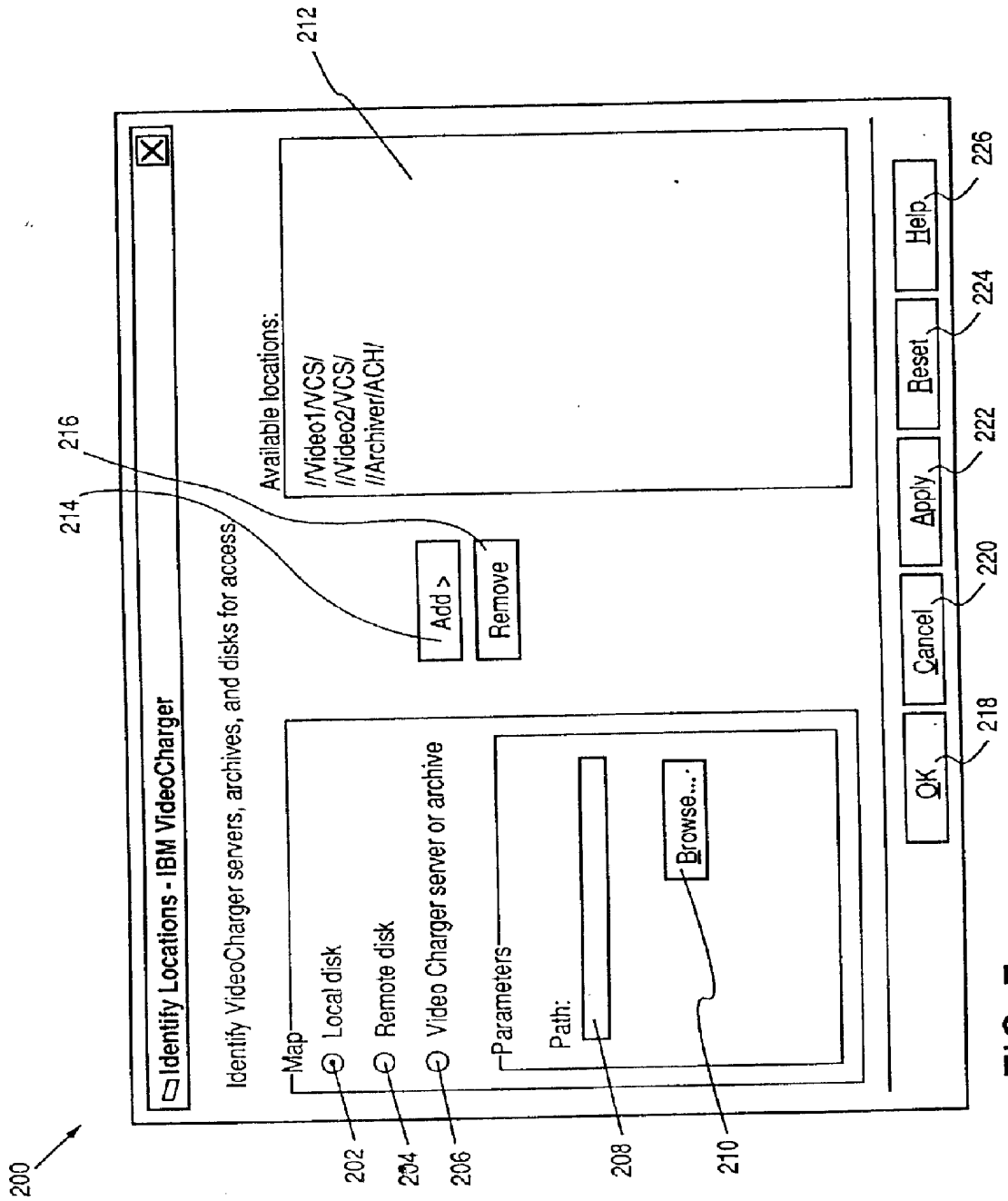

200
Identify Locations - IBM VideoCharger
Identify VideoCharger servers, archives, and disks for access.
Map
Local disk
Remote disk
Video Charger server or archive
Parameters
Path:
Browse...
Add >
Remove
Available locations:
//Video1/VCS/
//Video2/VCS/
//Archiver/ACH/
OK
Cancel
Apply
Reset
Help
202
204
206
208
210
212
214
216
218
220
222
224
226

FIG. 7a

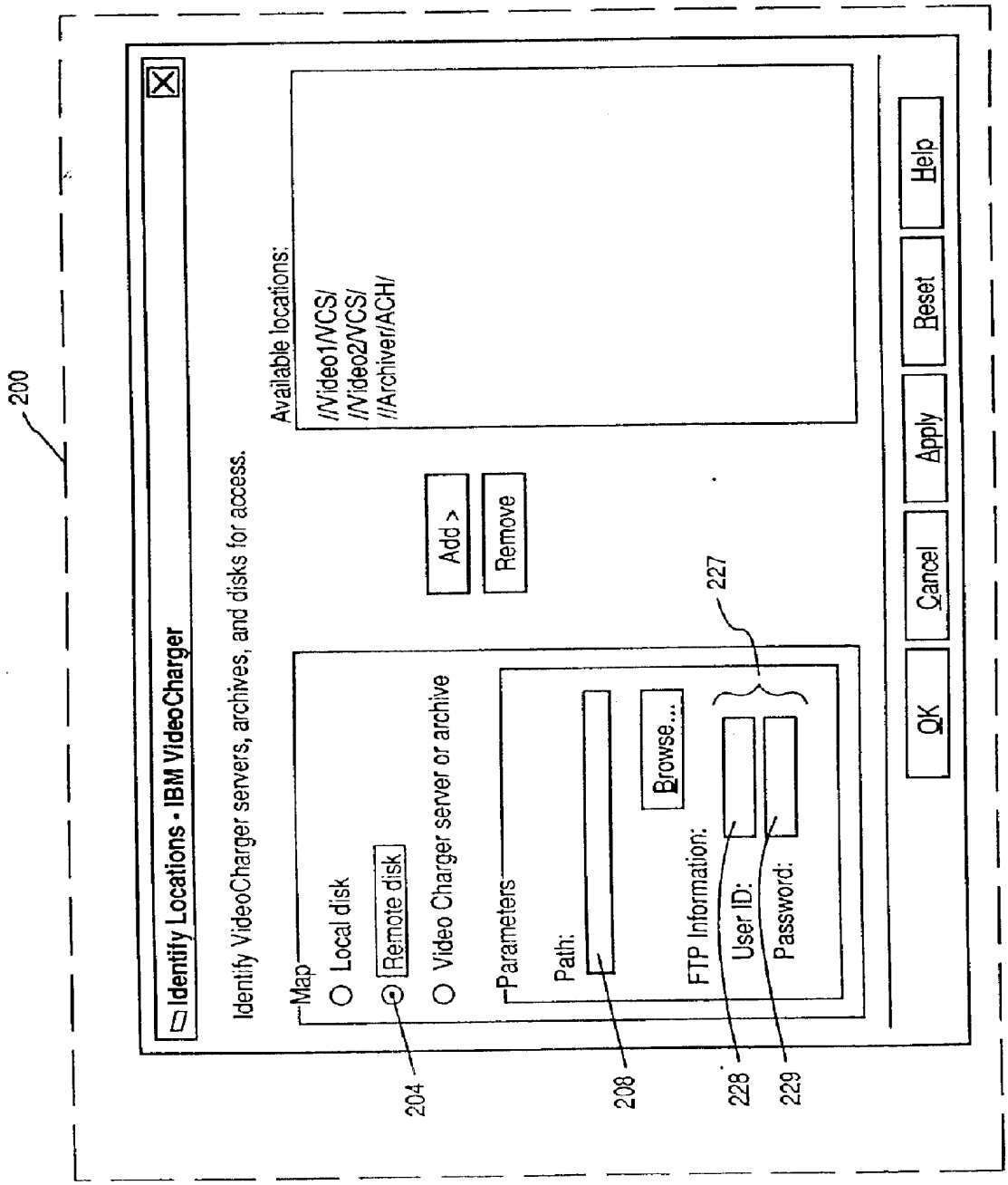

200
Identify Locations - IBM VideoCharger
Identify VideoCharger servers, archives, and disks for access.
Available locations:
//Video1/VCS/
//Video2/VCS/
//Archiver/ACH/
Add >
Remove
Map
Local disk
Remote disk
Video Charger server or archive
Parameters
Path:
Browse...
FTP Information:
User ID:
Password:
OK
Cancel
Apply
Reset
Help
204
208
227
228
229

FIG. 7b

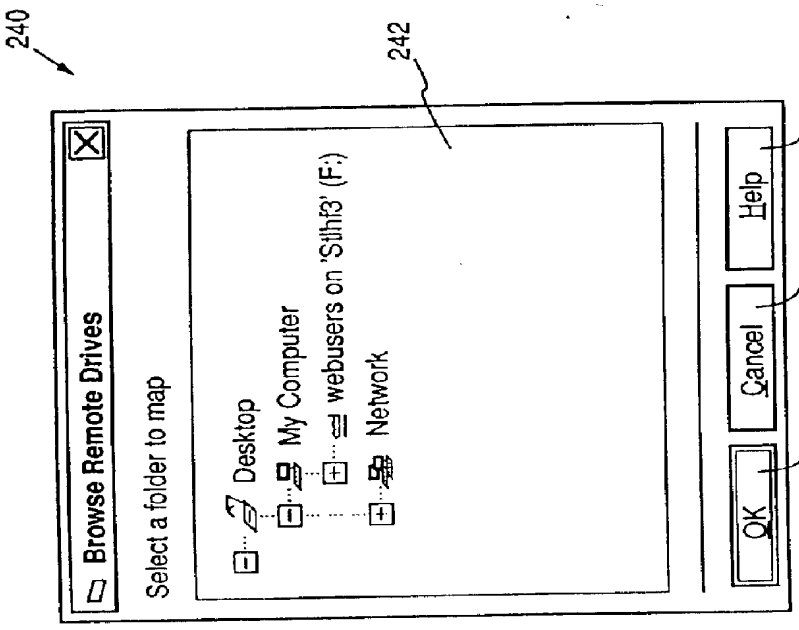

240
Browse Remote Drives
Select a folder to map
Desktop
My Computer
webusers on 'Stihl3' (F:)
Network
242
OK
Cancel
Help
244
246
248

FIG. 7d

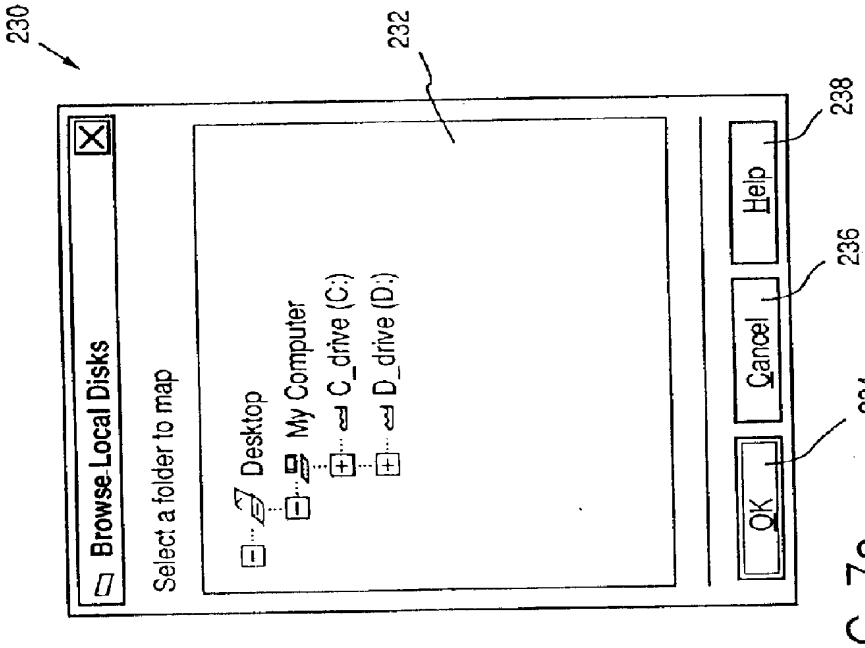

230
Browse Local Disks
Select a folder to map
Desktop
My Computer
C_drive (C:)
D_drive (D:)
232
OK
Cancel
Help
234
236
238

FIG. 7c

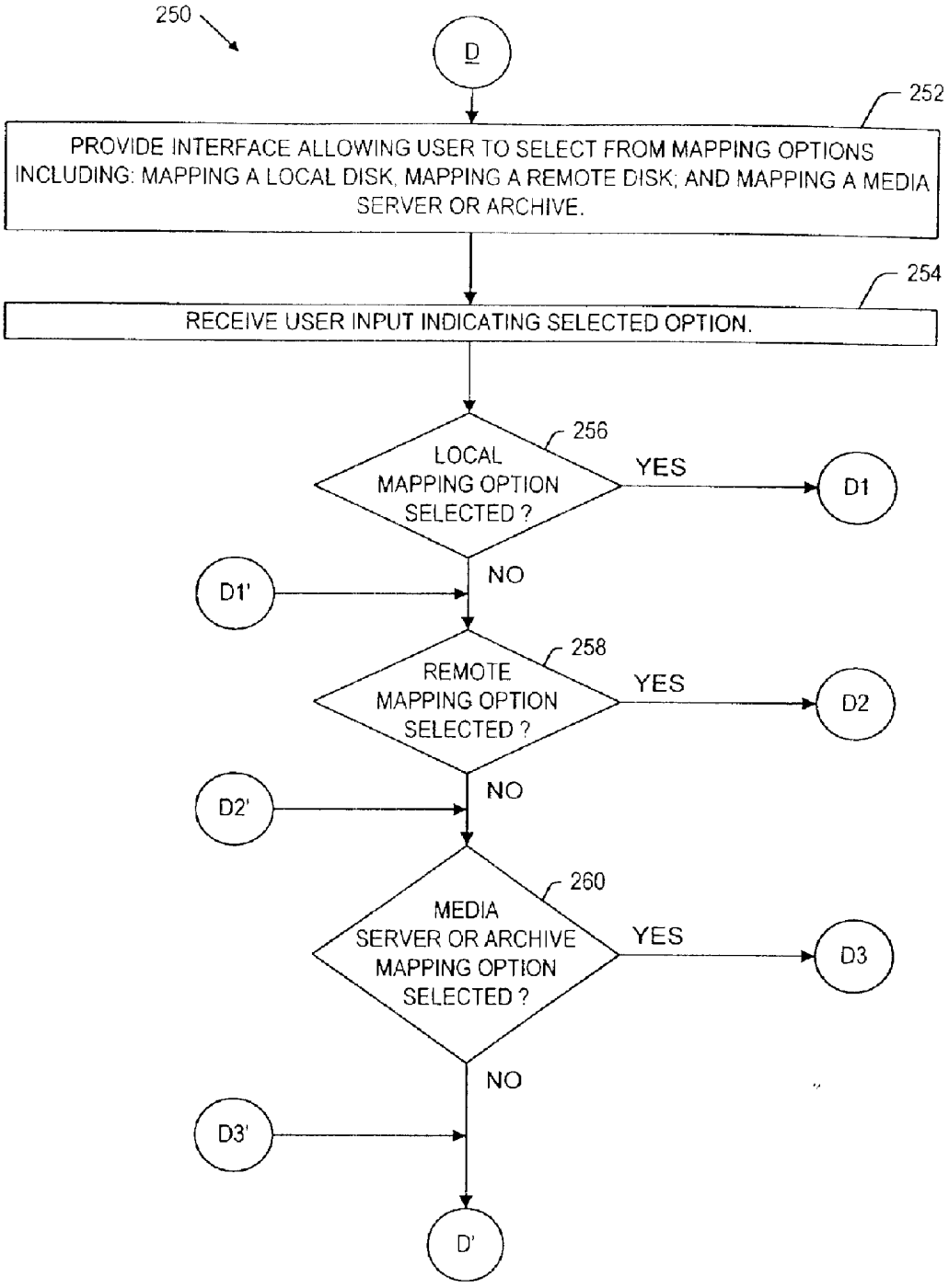

250
D
252
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM MAPPING OPTIONS INCLUDING: MAPPING A LOCAL DISK, MAPPING A REMOTE DISK; AND MAPPING A MEDIA SERVER OR ARCHIVE.
254
RECEIVE USER INPUT INDICATING SELECTED OPTION.
256
LOCAL MAPPING OPTION SELECTED ?
YES
D1
NO
D1'
258
REMOTE MAPPING OPTION SELECTED ?
YES
D2
NO
D2'
260
MEDIA SERVER OR ARCHIVE MAPPING OPTION SELECTED ?
YES
D3
NO
D3'
D'

FIG. 8A

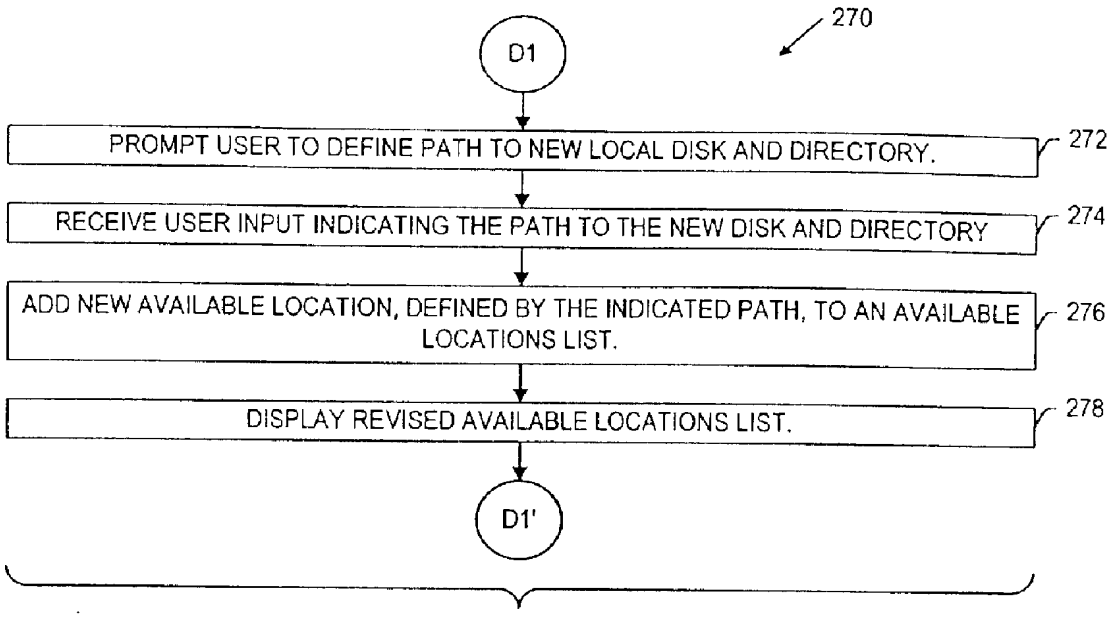

270
D1
PROMPT USER TO DEFINE PATH TO NEW LOCAL DISK AND DIRECTORY.
272
RECEIVE USER INPUT INDICATING THE PATH TO THE NEW DISK AND DIRECTORY
274
ADD NEW AVAILABLE LOCATION, DEFINED BY THE INDICATED PATH, TO AN AVAILABLE LOCATIONS LIST.
276
DISPLAY REVISED AVAILABLE LOCATIONS LIST.
278
D1'

FIG. 8B

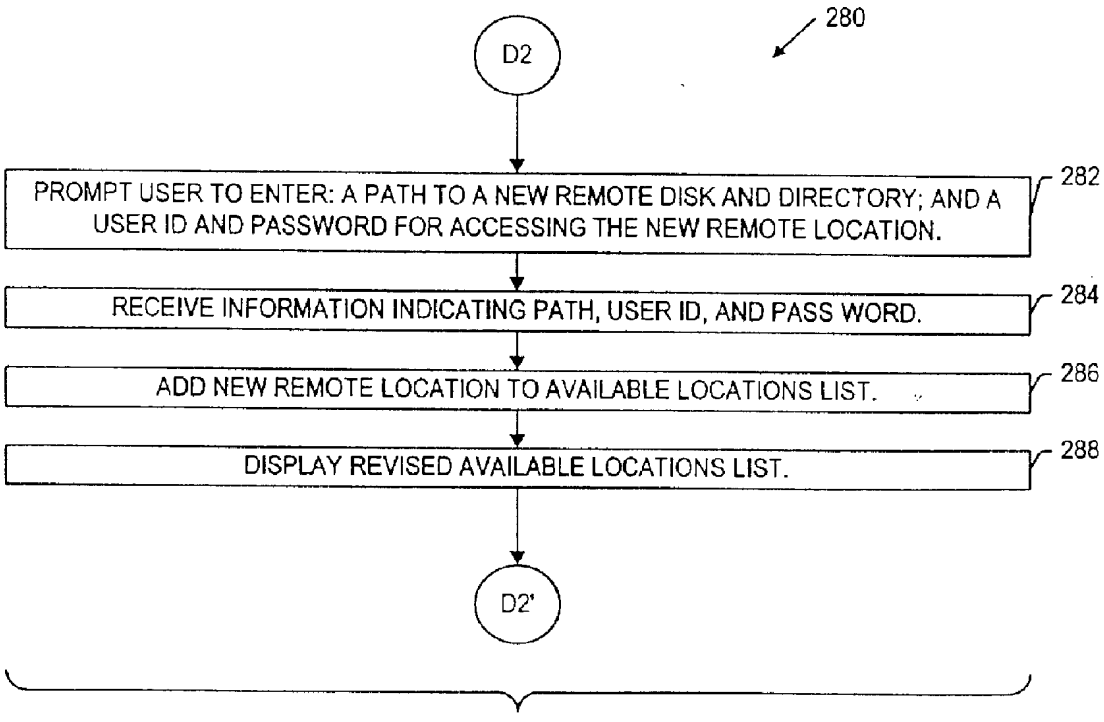

280
D2
PROMPT USER TO ENTER: A PATH TO A NEW REMOTE DISK AND DIRECTORY; AND A USER ID AND PASSWORD FOR ACCESSING THE NEW REMOTE LOCATION.
282
RECEIVE INFORMATION INDICATING PATH, USER ID, AND PASS WORD.
284
ADD NEW REMOTE LOCATION TO AVAILABLE LOCATIONS LIST.
286
DISPLAY REVISED AVAILABLE LOCATIONS LIST.
288
D2'

FIG. 8C

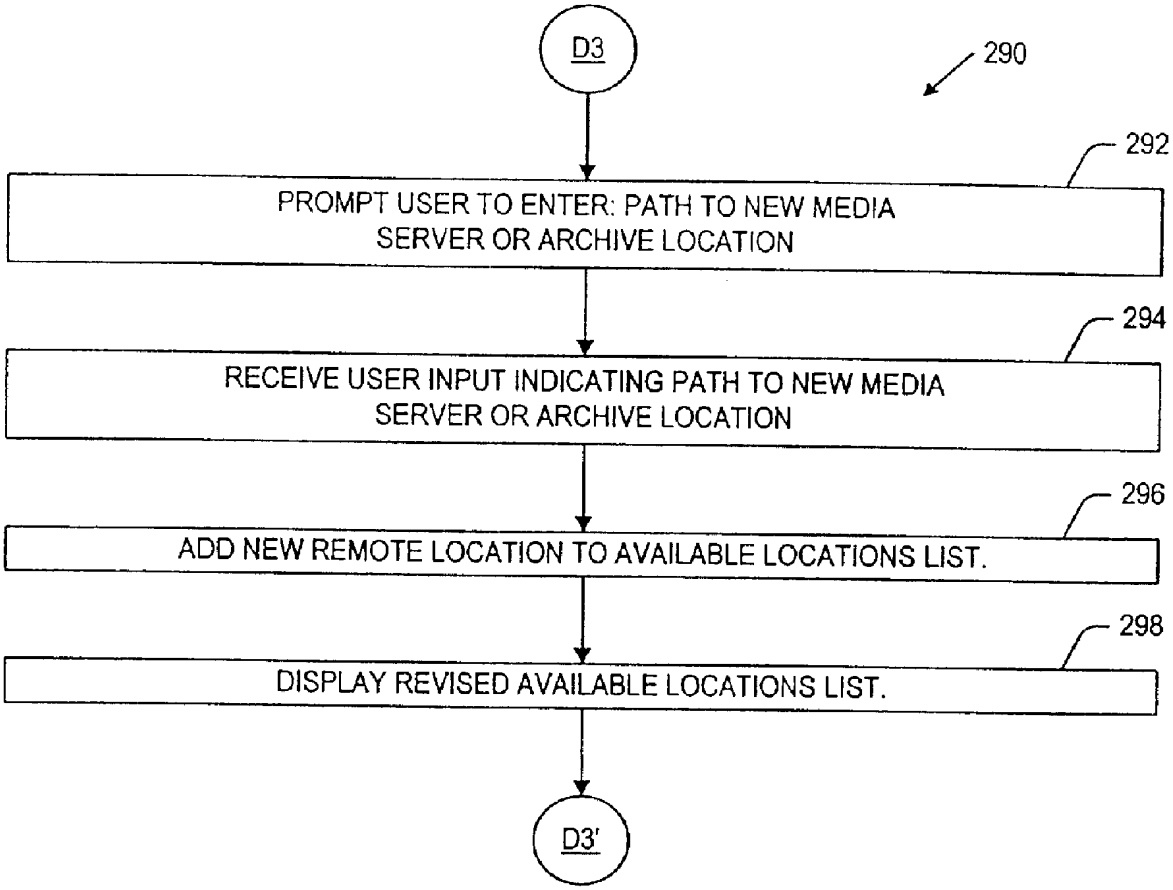

D3
290
292
PROMPT USER TO ENTER: PATH TO NEW MEDIA SERVER OR ARCHIVE LOCATION
294
RECEIVE USER INPUT INDICATING PATH TO NEW MEDIA SERVER OR ARCHIVE LOCATION
296
ADD NEW REMOTE LOCATION TO AVAILABLE LOCATIONS LIST.
298
DISPLAY REVISED AVAILABLE LOCATIONS LIST.
D3'

FIG. 8D

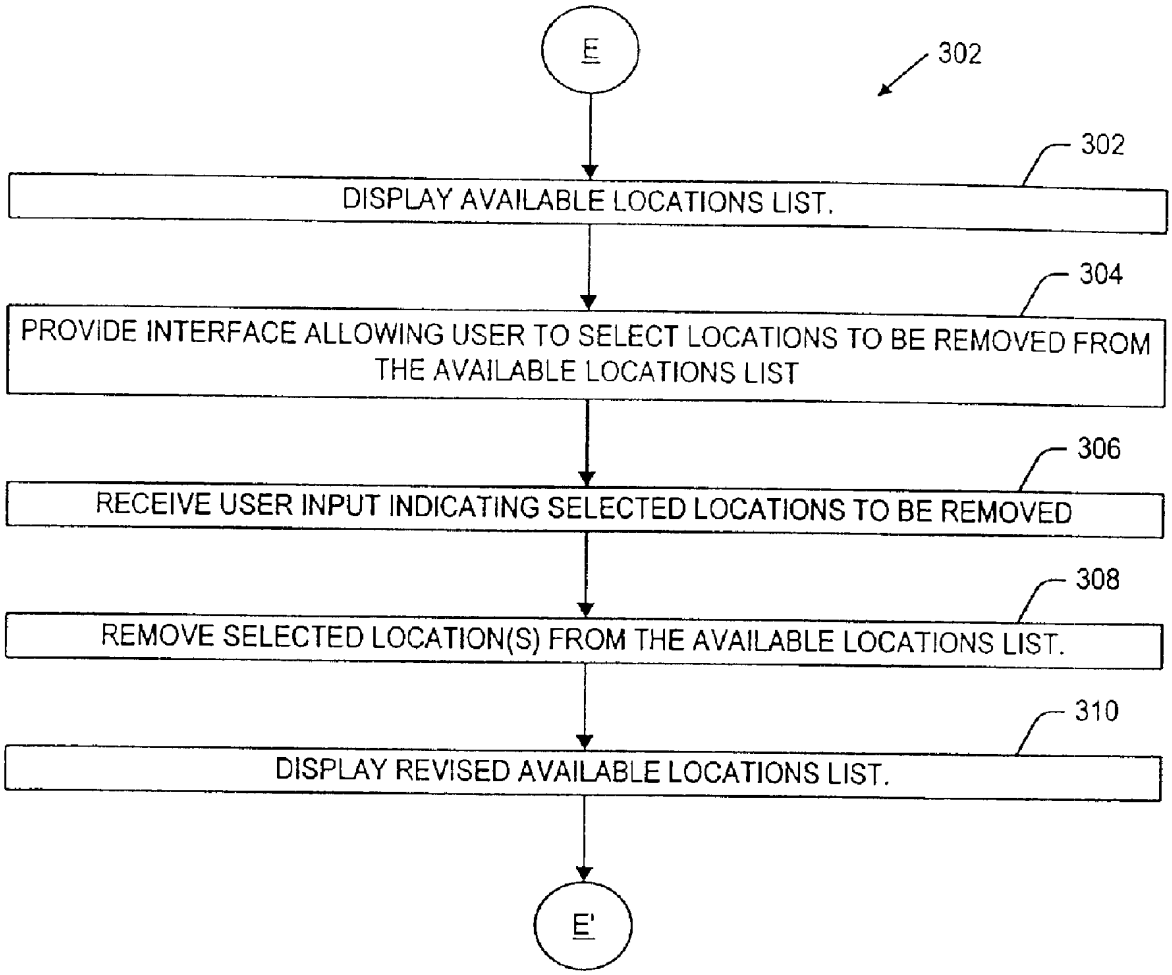

E
302
302
DISPLAY AVAILABLE LOCATIONS LIST.
304
PROVIDE INTERFACE ALLOWING USER TO SELECT LOCATIONS TO BE REMOVED FROM THE AVAILABLE LOCATIONS LIST
306
RECEIVE USER INPUT INDICATING SELECTED LOCATIONS TO BE REMOVED
308
REMOVE SELECTED LOCATION(S) FROM THE AVAILABLE LOCATIONS LIST.
310
DISPLAY REVISED AVAILABLE LOCATIONS LIST.
E'

FIG. 9

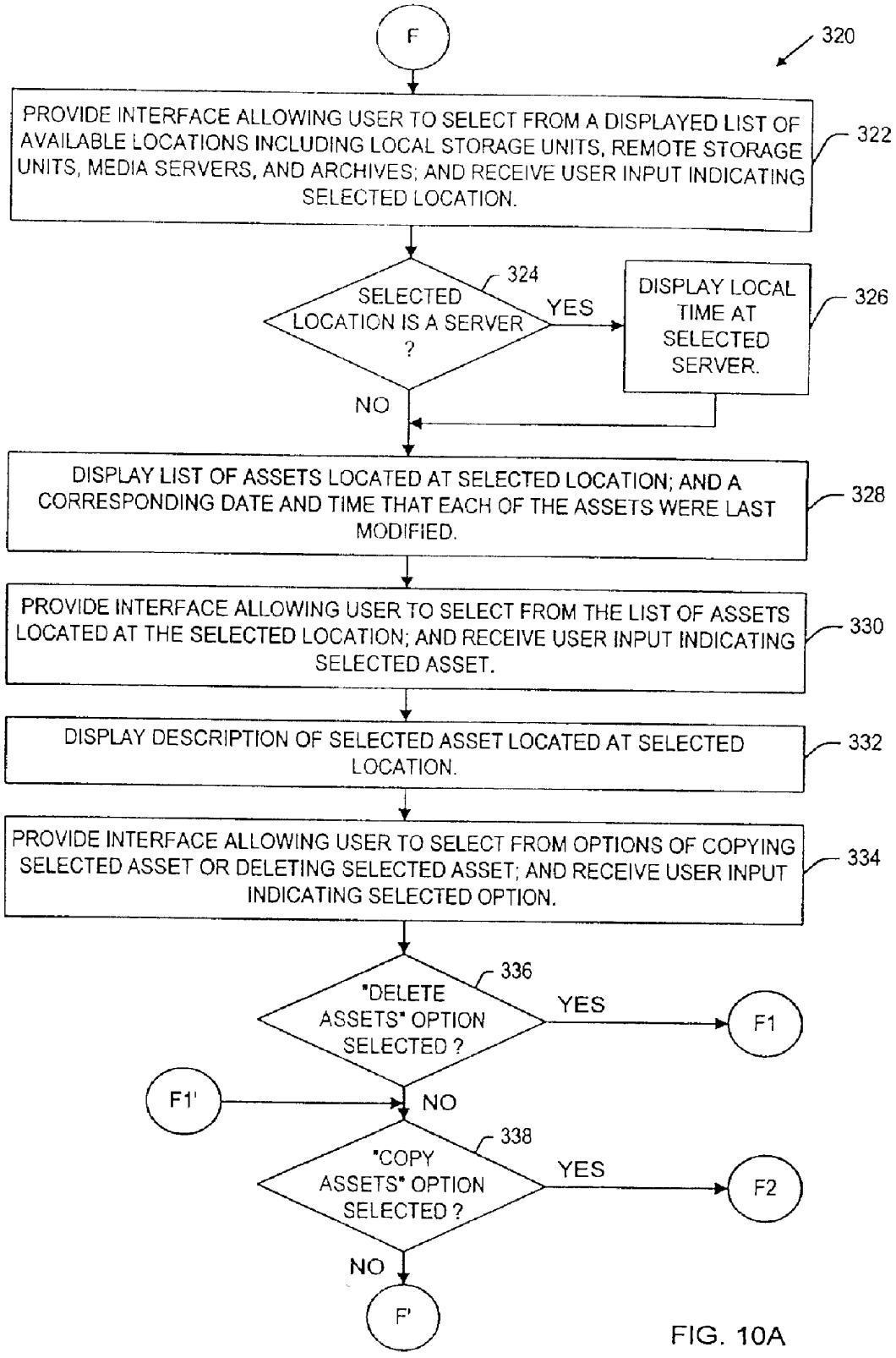

F
320
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A DISPLAYED LIST OF AVAILABLE LOCATIONS INCLUDING LOCAL STORAGE UNITS, REMOTE STORAGE UNITS, MEDIA SERVERS, AND ARCHIVES; AND RECEIVE USER INPUT INDICATING SELECTED LOCATION.
322
SELECTED LOCATION IS A SERVER ?
324
YES
DISPLAY LOCAL TIME AT SELECTED SERVER.
326
NO
DISPLAY LIST OF ASSETS LOCATED AT SELECTED LOCATION; AND A CORRESPONDING DATE AND TIME THAT EACH OF THE ASSETS WERE LAST MODIFIED.
328
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM THE LIST OF ASSETS LOCATED AT THE SELECTED LOCATION; AND RECEIVE USER INPUT INDICATING SELECTED ASSET.
330
DISPLAY DESCRIPTION OF SELECTED ASSET LOCATED AT SELECTED LOCATION.
332
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM OPTIONS OF COPYING SELECTED ASSET OR DELETING SELECTED ASSET; AND RECEIVE USER INPUT INDICATING SELECTED OPTION.
334
"DELETE ASSETS" OPTION SELECTED ?
336
YES
F1
F1'
NO
"COPY ASSETS" OPTION SELECTED ?
338
YES
F2
NO
F'

FIG. 10A

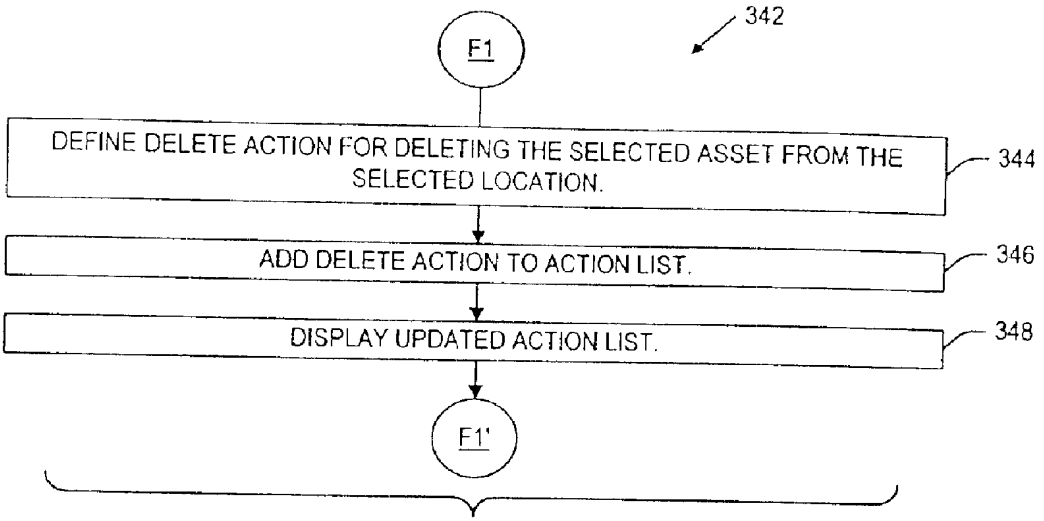


FIG. 10B

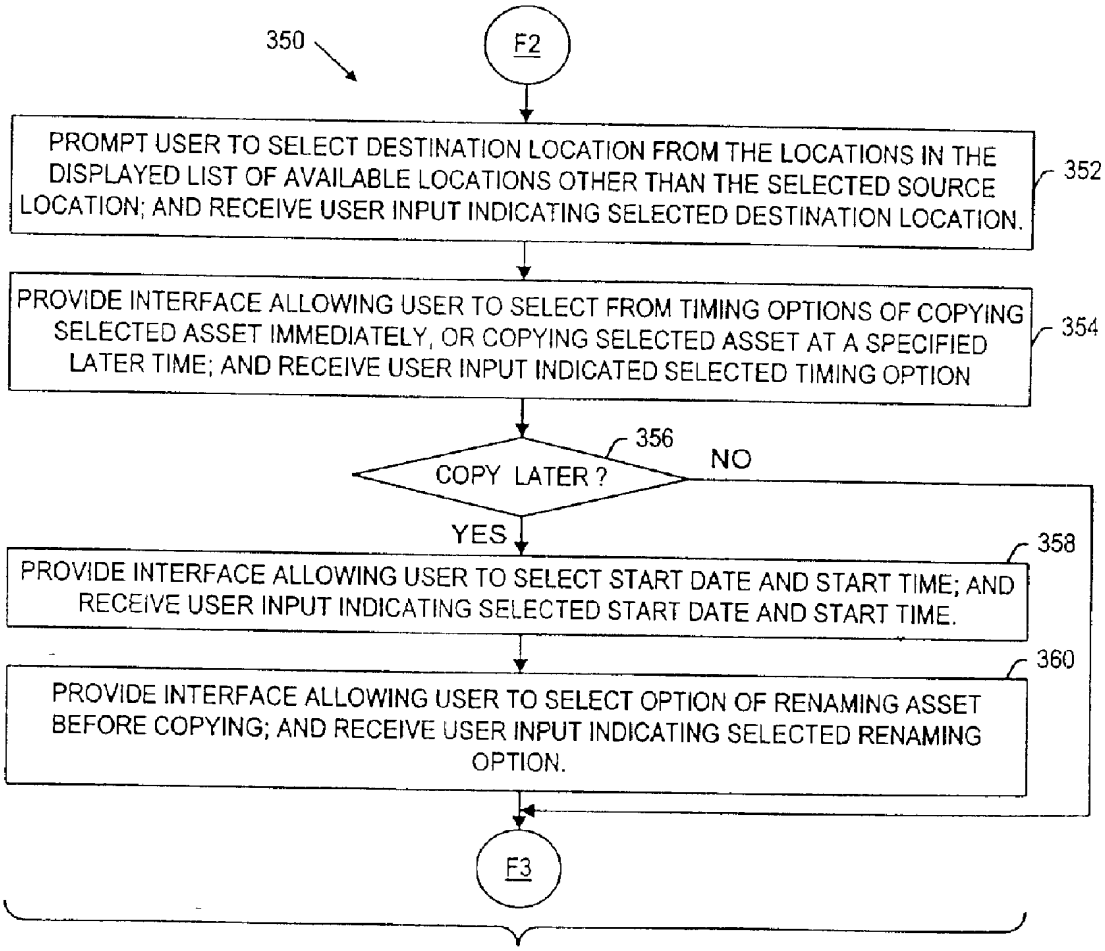


FIG. 10C

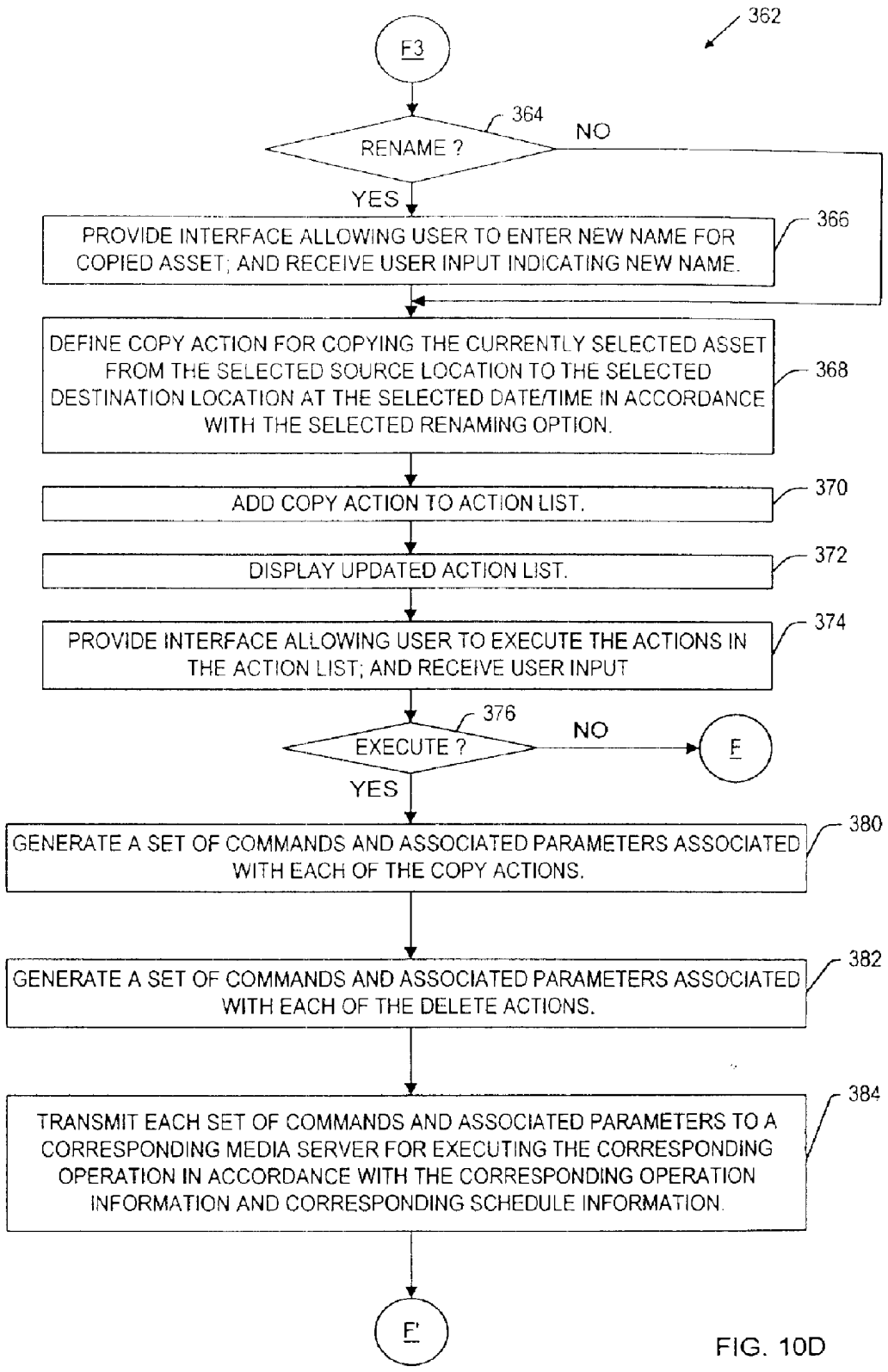

362
F3
364
RENAME ?
NO
YES
PROVIDE INTERFACE ALLOWING USER TO ENTER NEW NAME FOR COPIED ASSET; AND RECEIVE USER INPUT INDICATING NEW NAME.
366
DEFINE COPY ACTION FOR COPYING THE CURRENTLY SELECTED ASSET FROM THE SELECTED SOURCE LOCATION TO THE SELECTED DESTINATION LOCATION AT THE SELECTED DATE/TIME IN ACCORDANCE WITH THE SELECTED RENAMING OPTION.
368
ADD COPY ACTION TO ACTION LIST.
370
DISPLAY UPDATED ACTION LIST.
372
PROVIDE INTERFACE ALLOWING USER TO EXECUTE THE ACTIONS IN THE ACTION LIST; AND RECEIVE USER INPUT
374
376
EXECUTE ?
NO
F
YES
GENERATE A SET OF COMMANDS AND ASSOCIATED PARAMETERS ASSOCIATED WITH EACH OF THE COPY ACTIONS.
380
GENERATE A SET OF COMMANDS AND ASSOCIATED PARAMETERS ASSOCIATED WITH EACH OF THE DELETE ACTIONS.
382
TRANSMIT EACH SET OF COMMANDS AND ASSOCIATED PARAMETERS TO A CORRESPONDING MEDIA SERVER FOR EXECUTING THE CORRESPONDING OPERATION IN ACCORDANCE WITH THE CORRESPONDING OPERATION INFORMATION AND CORRESPONDING SCHEDULE INFORMATION.
384
F'

FIG. 10D

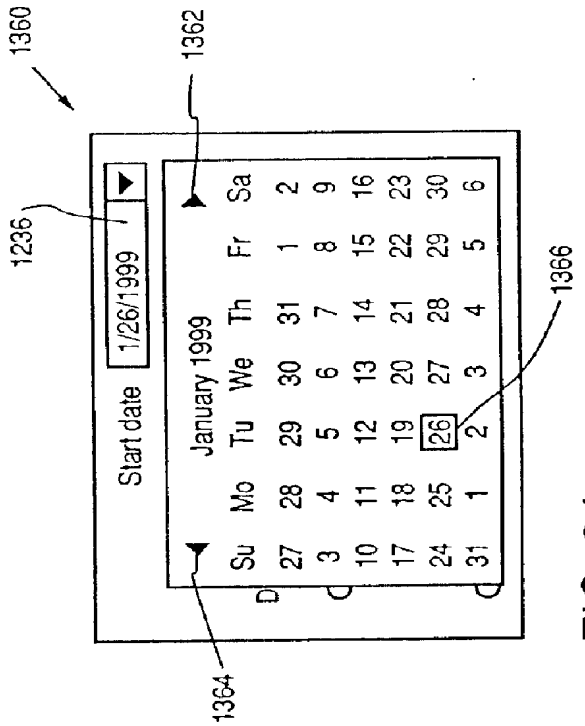

1360
1362
1236
Start date
1/26/1999
January 1999
Su Mo Tu We Th Fr Sa
1366
1364

FIG. 34

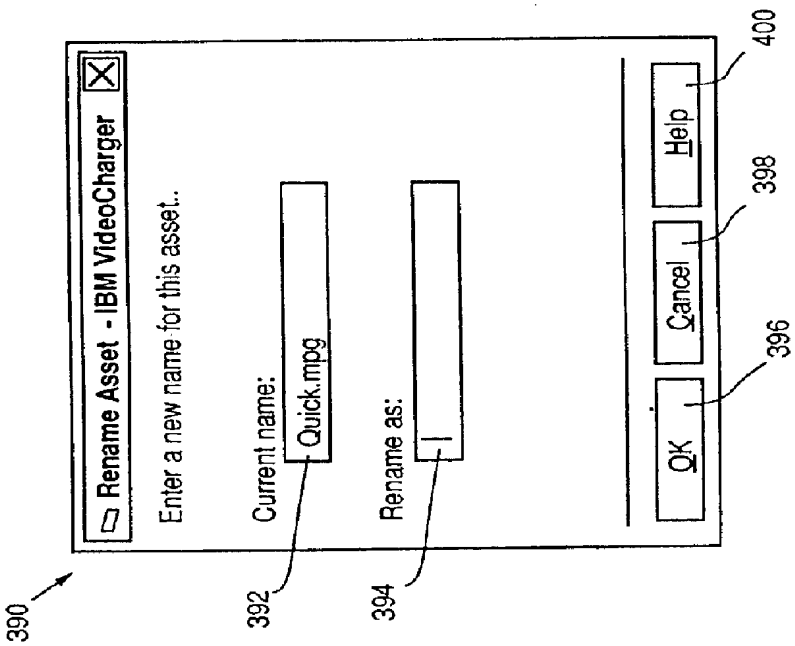

Rename Asset - IBM VideoCharger
Enter a new name for this asset..
Current name:
Quick.mpg
Rename as:
OK
Cancel
Help
390
392
394
396
398
400

FIG. 11

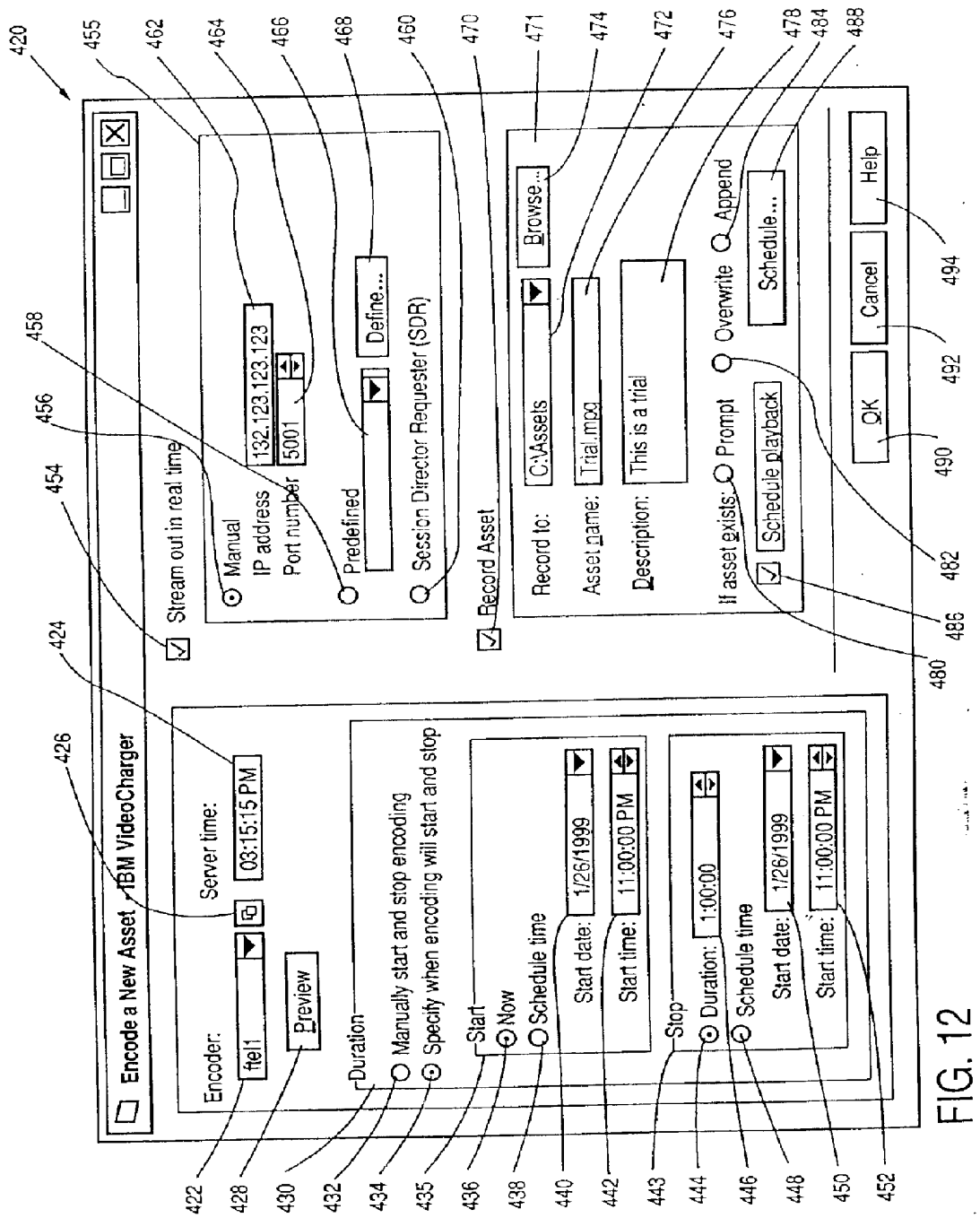

Encode a New Asset - IBM VideoCharger
Encoder:
fte1
Preview
Server time:
08:15:15 PM
Stream out in real time
Manual
IP address
132.123.123.123
Port number
5001
Predefined
Define...
Session Director Requester (SDR)
Record Asset
Record to:
C:\Assets
Browse...
Asset name:
Trial.mpg
Description:
This is a trial
If asset exists:
Prompt
Overwrite
Append
Schedule playback
Schedule...
OK
Cancel
Help
Duration
Manually start and stop encoding
Specify when encoding will start and stop
Start
Now
Schedule time
Start date:
1/26/1999
Start time:
11:00:00 PM
Stop
Duration:
1:00:00
Schedule time
Start date:
1/26/1999
Start time:
11:00:00 PM
420
422
424
426
428
430
432
434
435
436
438
440
442
443
444
446
448
450
452
454
455
456
458
460
462
464
466
468
470
471
472
474
476
478
480
482
484
486
488
490
492
494

FIG. 12

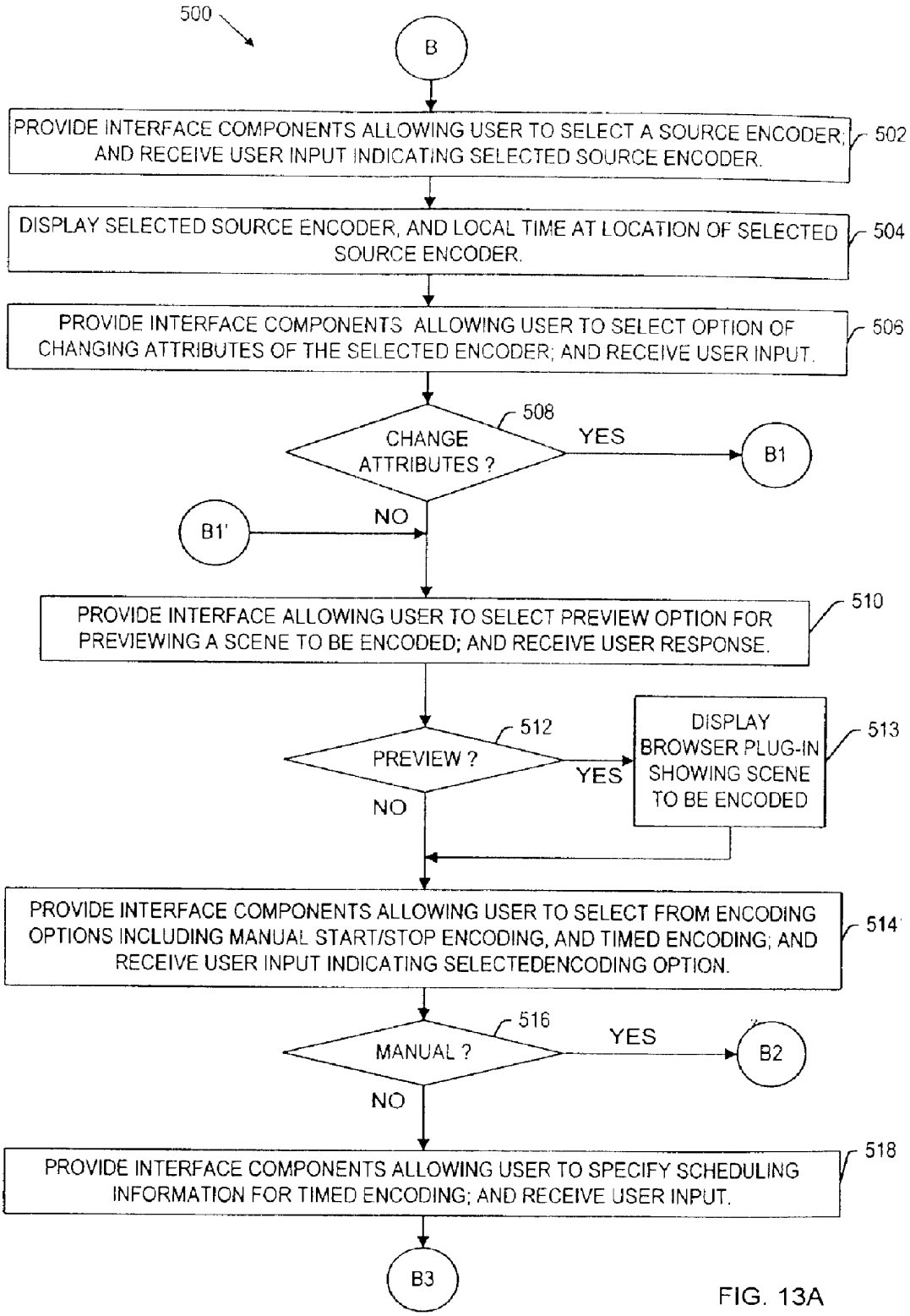

500
B
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT A SOURCE ENCODER; AND RECEIVE USER INPUT INDICATING SELECTED SOURCE ENCODER.
502
DISPLAY SELECTED SOURCE ENCODER, AND LOCAL TIME AT LOCATION OF SELECTED SOURCE ENCODER.
504
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT OPTION OF CHANGING ATTRIBUTES OF THE SELECTED ENCODER; AND RECEIVE USER INPUT.
506
508
CHANGE ATTRIBUTES ?
YES
B1
B1'
NO
PROVIDE INTERFACE ALLOWING USER TO SELECT PREVIEW OPTION FOR PREVIEWING A SCENE TO BE ENCODED; AND RECEIVE USER RESPONSE.
510
512
PREVIEW ?
YES
DISPLAY BROWSER PLUG-IN SHOWING SCENE TO BE ENCODED
513
NO
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT FROM ENCODING OPTIONS INCLUDING MANUAL START/STOP ENCODING, AND TIMED ENCODING; AND RECEIVE USER INPUT INDICATING SELECTEDENCODING OPTION.
514
516
MANUAL ?
YES
B2
NO
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SPECIFY SCHEDULING INFORMATION FOR TIMED ENCODING; AND RECEIVE USER INPUT.
518
B3

FIG. 13A

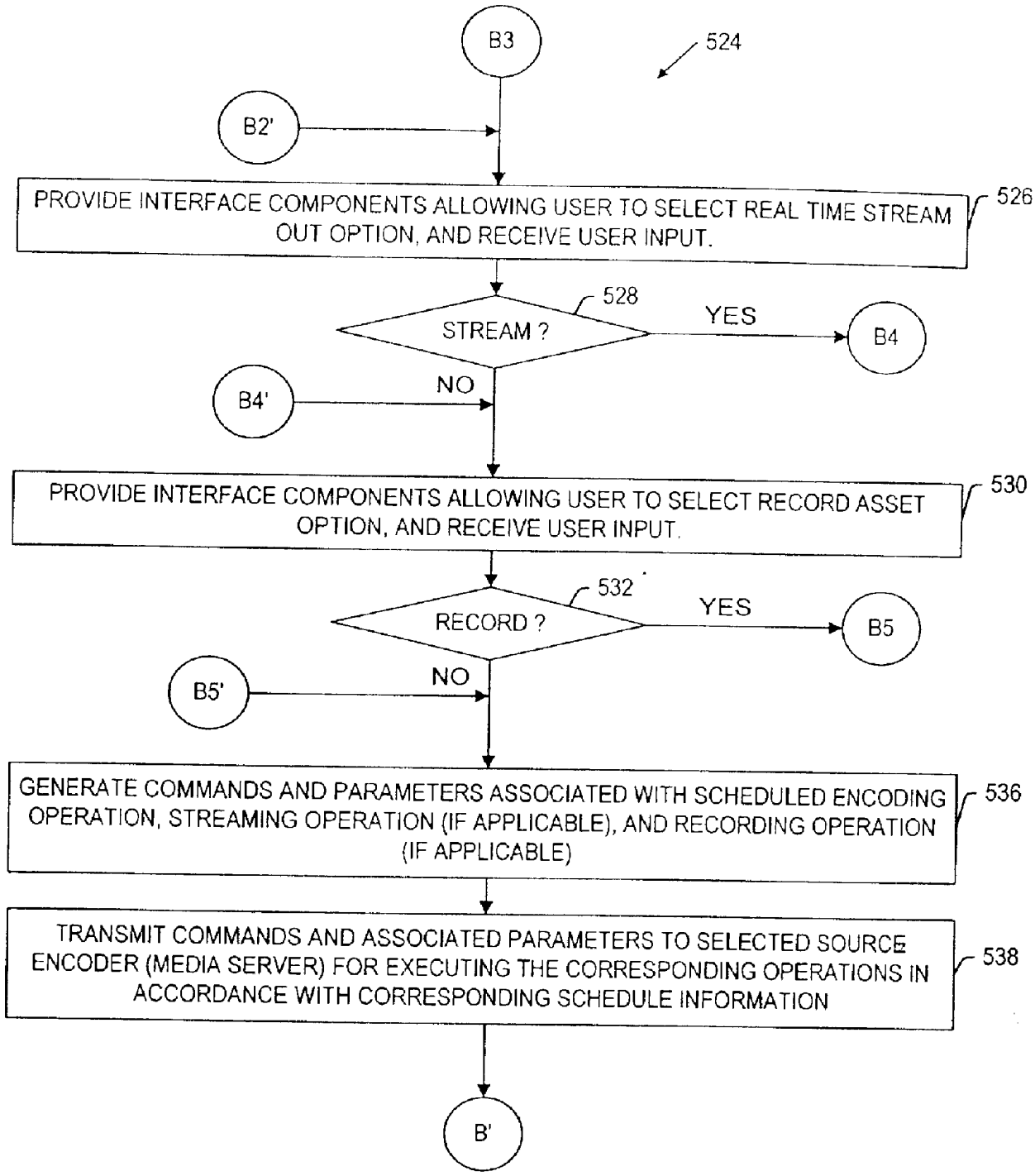

B3
524
B2'
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT REAL TIME STREAM OUT OPTION, AND RECEIVE USER INPUT.
526
528
STREAM ?
YES
B4
B4'
NO
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT RECORD ASSET OPTION, AND RECEIVE USER INPUT.
530
532
RECORD ?
YES
B5
B5'
NO
GENERATE COMMANDS AND PARAMETERS ASSOCIATED WITH SCHEDULED ENCODING OPERATION, STREAMING OPERATION (IF APPLICABLE), AND RECORDING OPERATION (IF APPLICABLE)
536
TRANSMIT COMMANDS AND ASSOCIATED PARAMETERS TO SELECTED SOURCE ENCODER (MEDIA SERVER) FOR EXECUTING THE CORRESPONDING OPERATIONS IN ACCORDANCE WITH CORRESPONDING SCHEDULE INFORMATION
538
B'

FIG. 13B

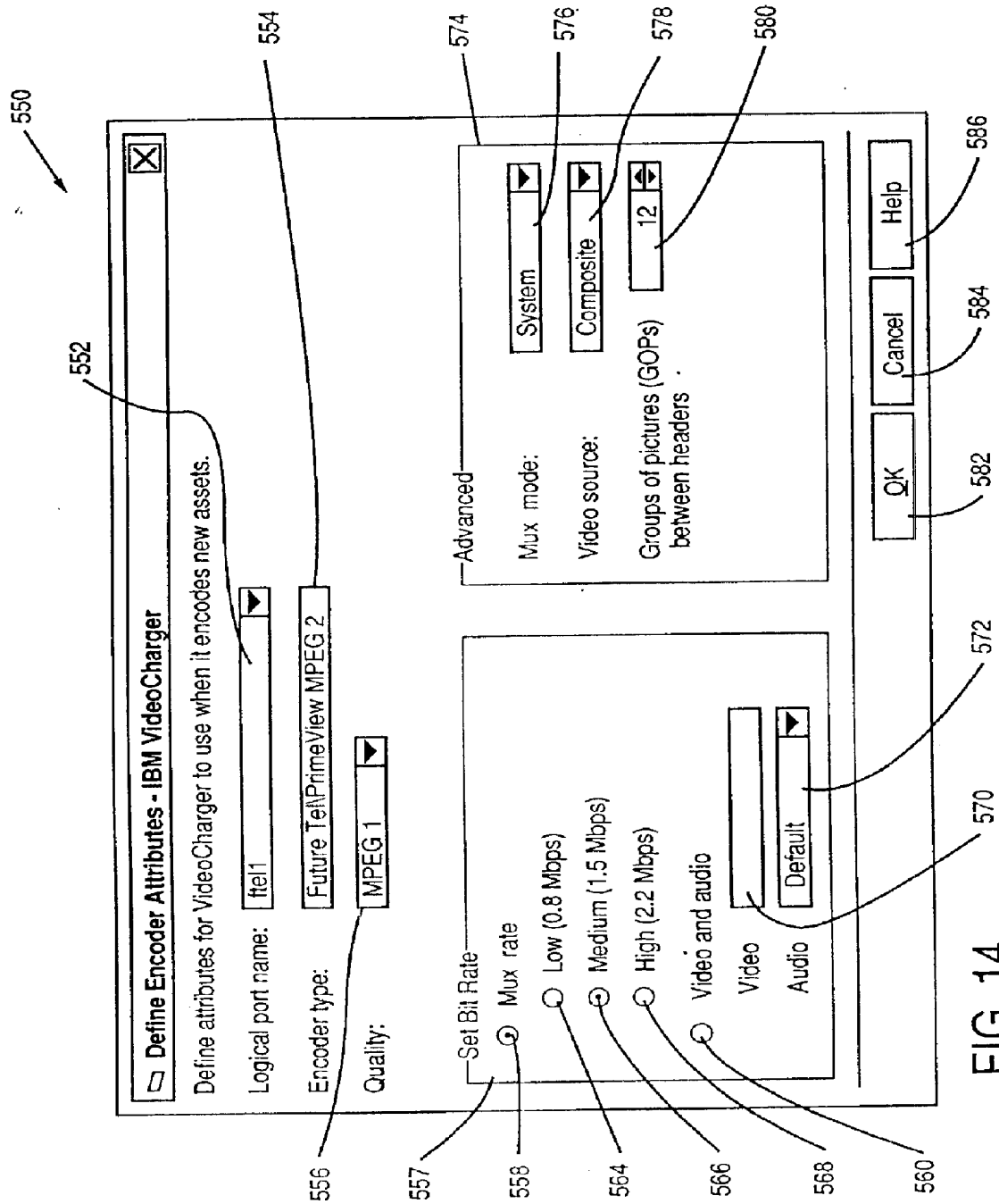

550
552
554
556
557
558
560
564
566
568
570
572
574
576
578
580
582
584
586
Define Encoder Attributes - IBM VideoCharger
Define attributes for VideoCharger to use when it encodes new assets.
Logical port name:
Encoder type:
Quality:
Set Bit Rate
Mux rate
Low (0.8 Mbps)
Medium (1.5 Mbps)
High (2.2 Mbps)
Video and audio
Video
Audio
Default
Advanced
Mux mode:
Video source:
Groups of pictures (GOPs) between headers
System
Composite
12
OK
Cancel
Help

FIG. 14

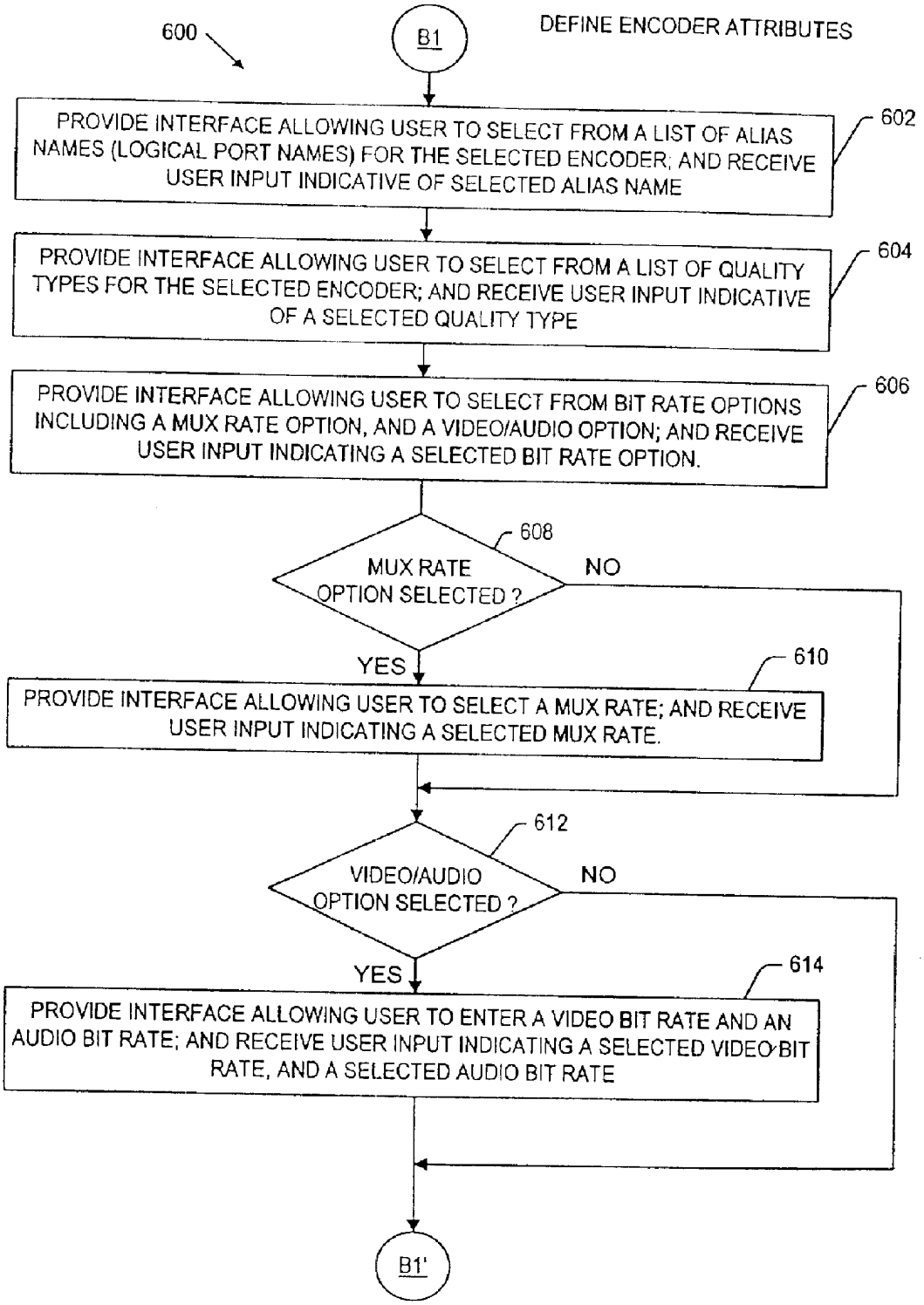

600
B1
DEFINE ENCODER ATTRIBUTES
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A LIST OF ALIAS NAMES (LOGICAL PORT NAMES) FOR THE SELECTED ENCODER; AND RECEIVE USER INPUT INDICATIVE OF SELECTED ALIAS NAME
602
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A LIST OF QUALITY TYPES FOR THE SELECTED ENCODER; AND RECEIVE USER INPUT INDICATIVE OF A SELECTED QUALITY TYPE
604
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM BIT RATE OPTIONS INCLUDING A MUX RATE OPTION, AND A VIDEO/AUDIO OPTION; AND RECEIVE USER INPUT INDICATING A SELECTED BIT RATE OPTION.
606
608
MUX RATE OPTION SELECTED ?
NO
YES
610
PROVIDE INTERFACE ALLOWING USER TO SELECT A MUX RATE; AND RECEIVE USER INPUT INDICATING A SELECTED MUX RATE.
612
VIDEO/AUDIO OPTION SELECTED ?
NO
YES
614
PROVIDE INTERFACE ALLOWING USER TO ENTER A VIDEO BIT RATE AND AN AUDIO BIT RATE; AND RECEIVE USER INPUT INDICATING A SELECTED VIDEO BIT RATE, AND A SELECTED AUDIO BIT RATE
B1'

FIG. 15

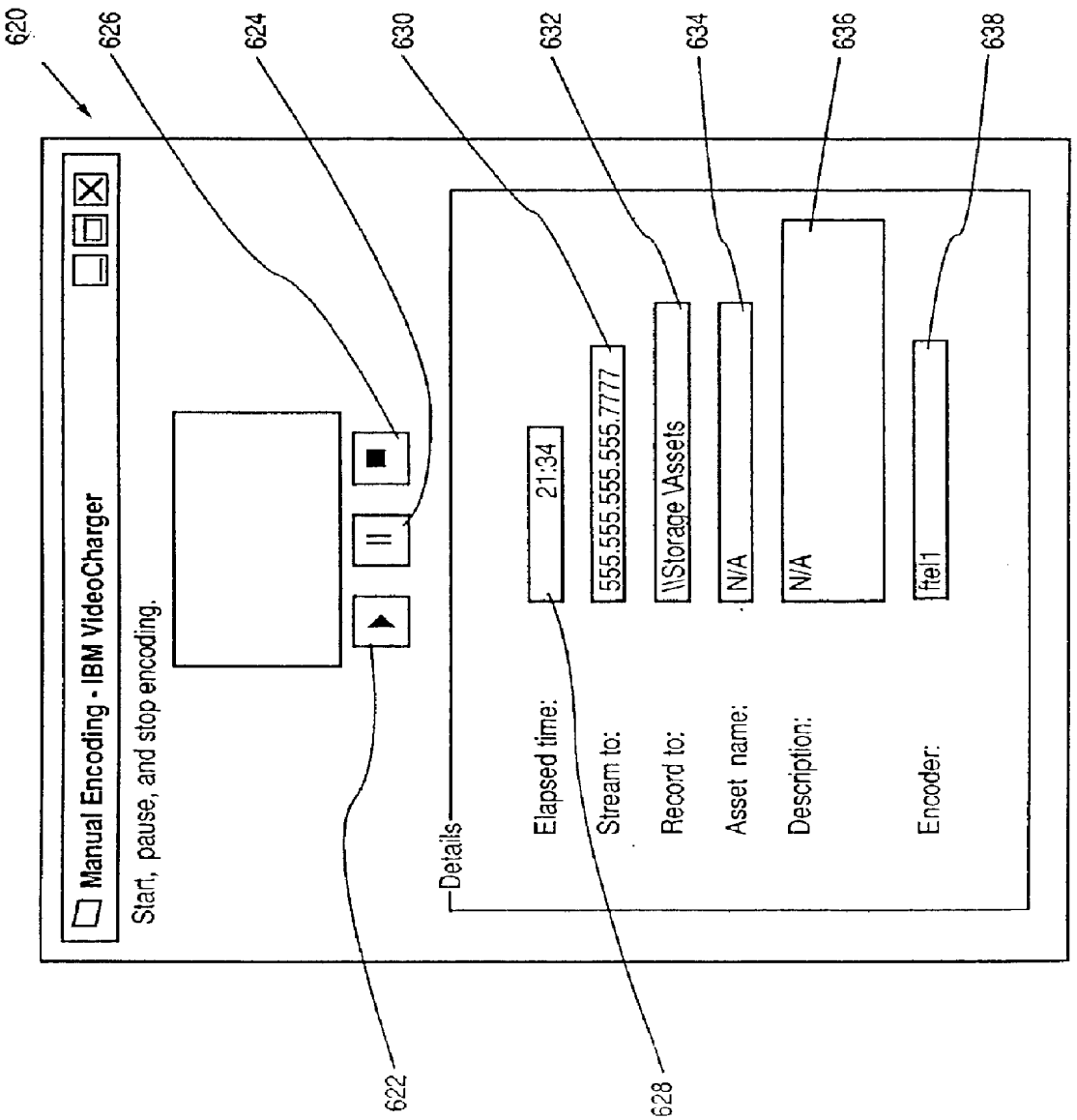

620
626
624
630
632
634
636
638
Manual Encoding - IBM VideoCharger
Start, pause, and stop encoding.
Details
Elapsed time:
21:34
Stream to:
555.555.555.555.7777
Record to:
\\Storage \Assets
Asset name:
N/A
Description:
N/A
Encoder:
ftel1
622
628

FIG. 16

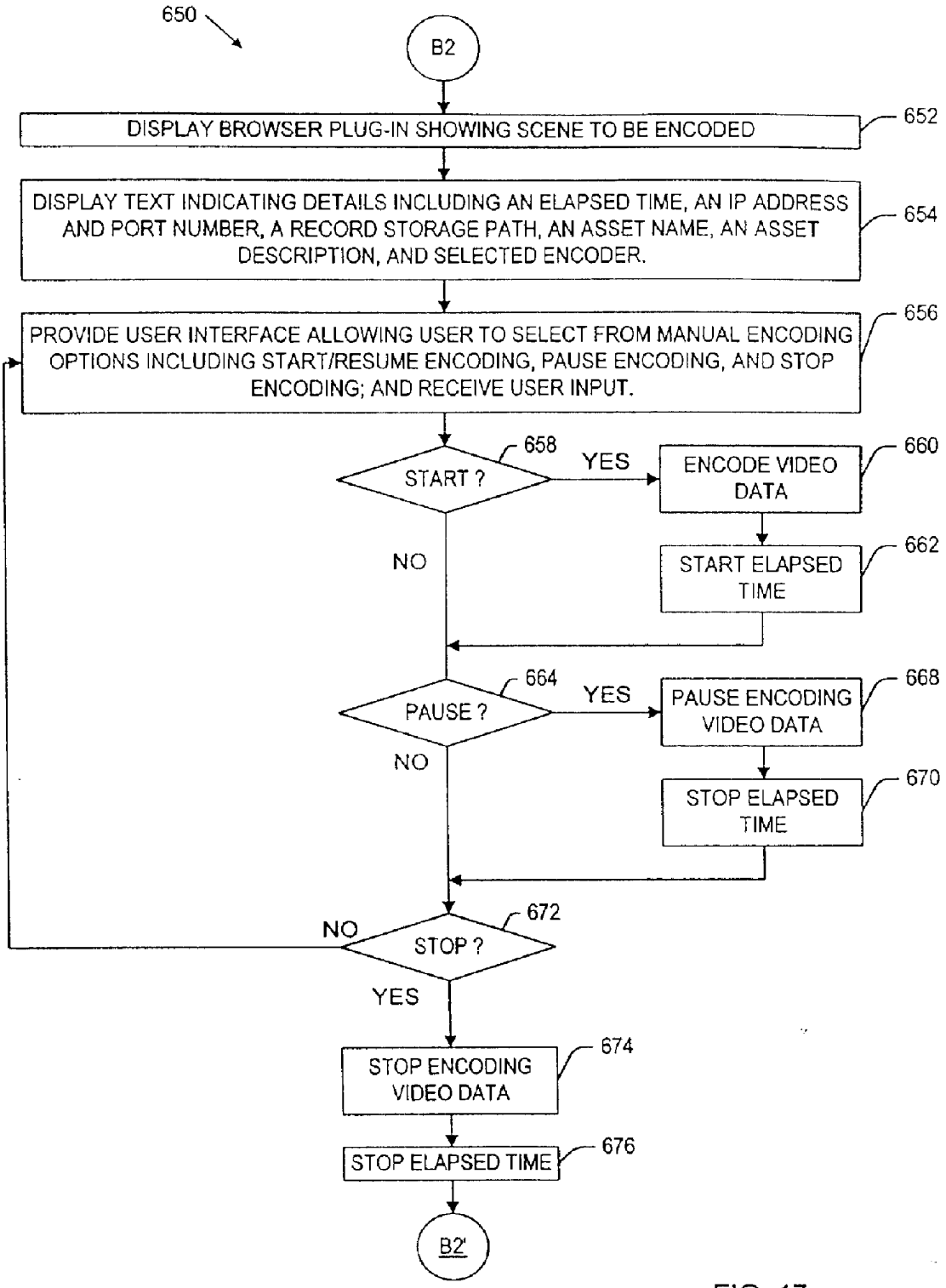

650
B2
DISPLAY BROWSER PLUG-IN SHOWING SCENE TO BE ENCODED
652
DISPLAY TEXT INDICATING DETAILS INCLUDING AN ELAPSED TIME, AN IP ADDRESS AND PORT NUMBER, A RECORD STORAGE PATH, AN ASSET NAME, AN ASSET DESCRIPTION, AND SELECTED ENCODER.
654
PROVIDE USER INTERFACE ALLOWING USER TO SELECT FROM MANUAL ENCODING OPTIONS INCLUDING START/RESUME ENCODING, PAUSE ENCODING, AND STOP ENCODING; AND RECEIVE USER INPUT.
656
START ?
658
YES
ENCODE VIDEO DATA
660
NO
START ELAPSED TIME
662
PAUSE ?
664
YES
PAUSE ENCODING VIDEO DATA
668
NO
STOP ELAPSED TIME
670
NO
STOP ?
672
YES
STOP ENCODING VIDEO DATA
674
STOP ELAPSED TIME
676
B2'

FIG. 17

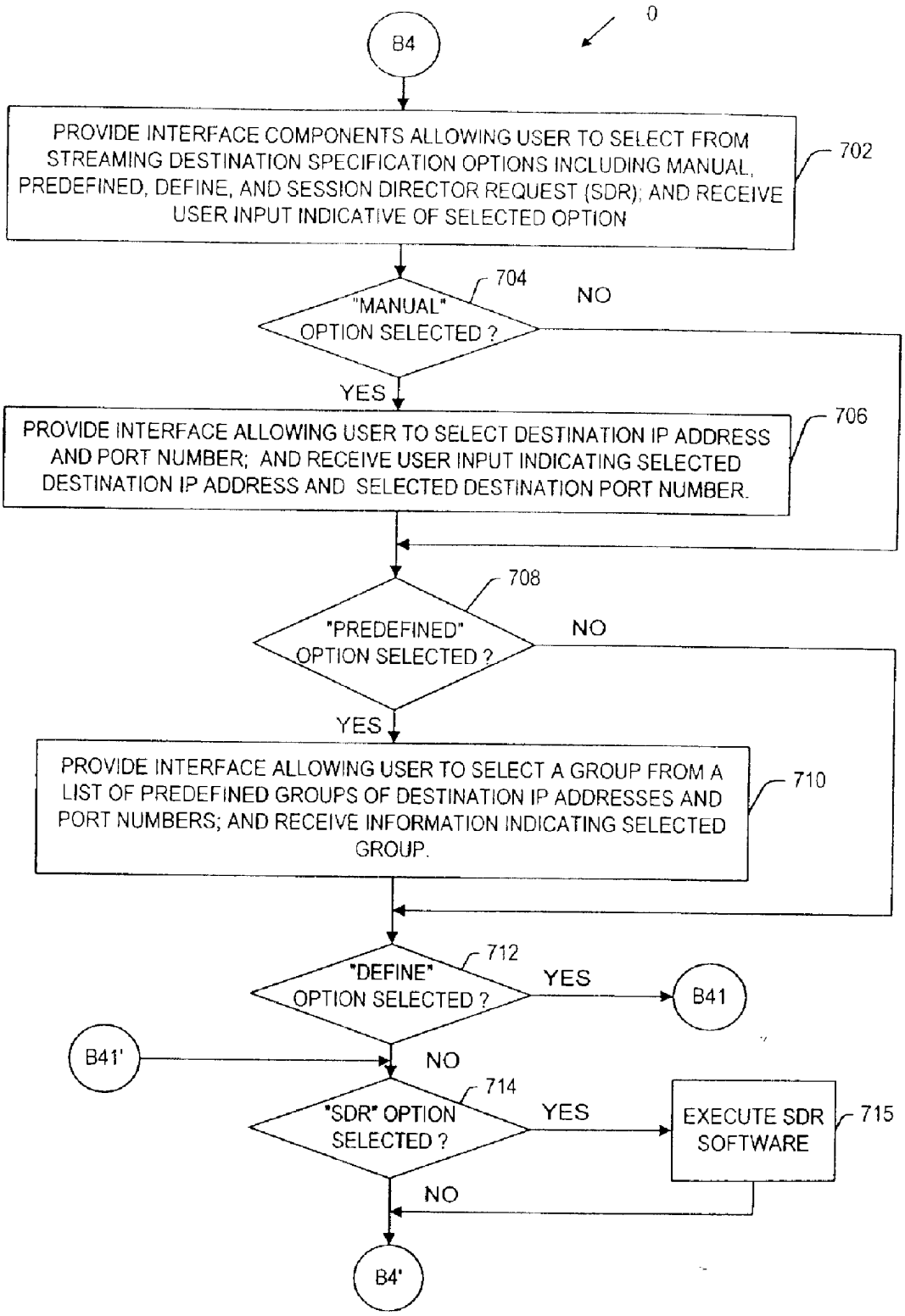

0
B4
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT FROM STREAMING DESTINATION SPECIFICATION OPTIONS INCLUDING MANUAL, PREDEFINED, DEFINE, AND SESSION DIRECTOR REQUEST (SDR); AND RECEIVE USER INPUT INDICATIVE OF SELECTED OPTION
702
"MANUAL" OPTION SELECTED ?
704
NO
YES
PROVIDE INTERFACE ALLOWING USER TO SELECT DESTINATION IP ADDRESS AND PORT NUMBER; AND RECEIVE USER INPUT INDICATING SELECTED DESTINATION IP ADDRESS AND SELECTED DESTINATION PORT NUMBER.
706
"PREDEFINED" OPTION SELECTED ?
708
NO
YES
PROVIDE INTERFACE ALLOWING USER TO SELECT A GROUP FROM A LIST OF PREDEFINED GROUPS OF DESTINATION IP ADDRESSES AND PORT NUMBERS; AND RECEIVE INFORMATION INDICATING SELECTED GROUP.
710
"DEFINE" OPTION SELECTED ?
712
YES
B41
B41'
NO
"SDR" OPTION SELECTED ?
714
YES
EXECUTE SDR SOFTWARE
715
NO
B4'

FIG. 18

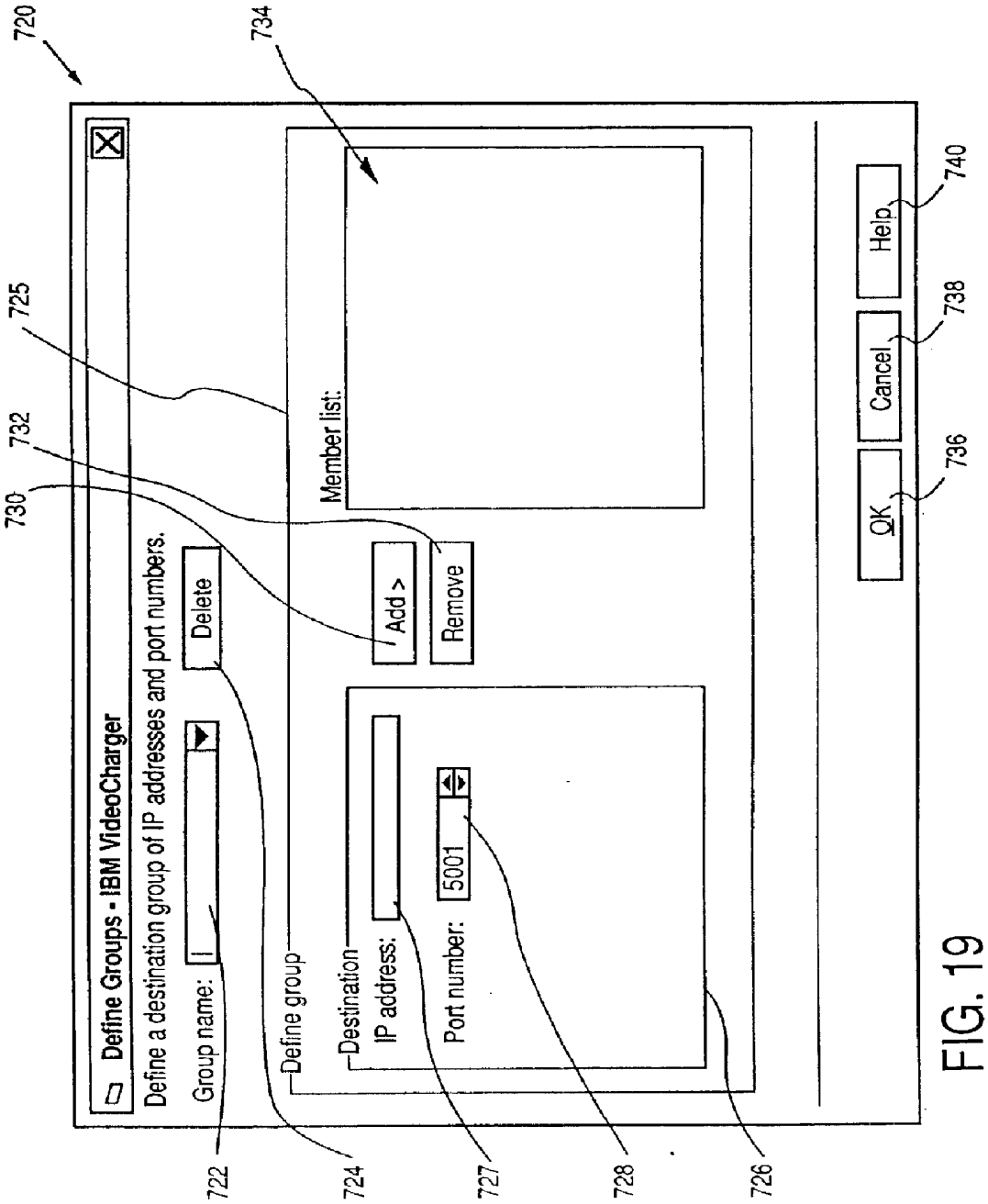

720
734
725
732
730
Define Groups - IBM VideoCharger
Define a destination group of IP addresses and port numbers.
Group name:
Delete
Add >
Remove
Member list:
Define group
Destination
IP address:
Port number:
5001
OK
Cancel
Help
740
738
736
722
724
727
728
726

FIG. 19

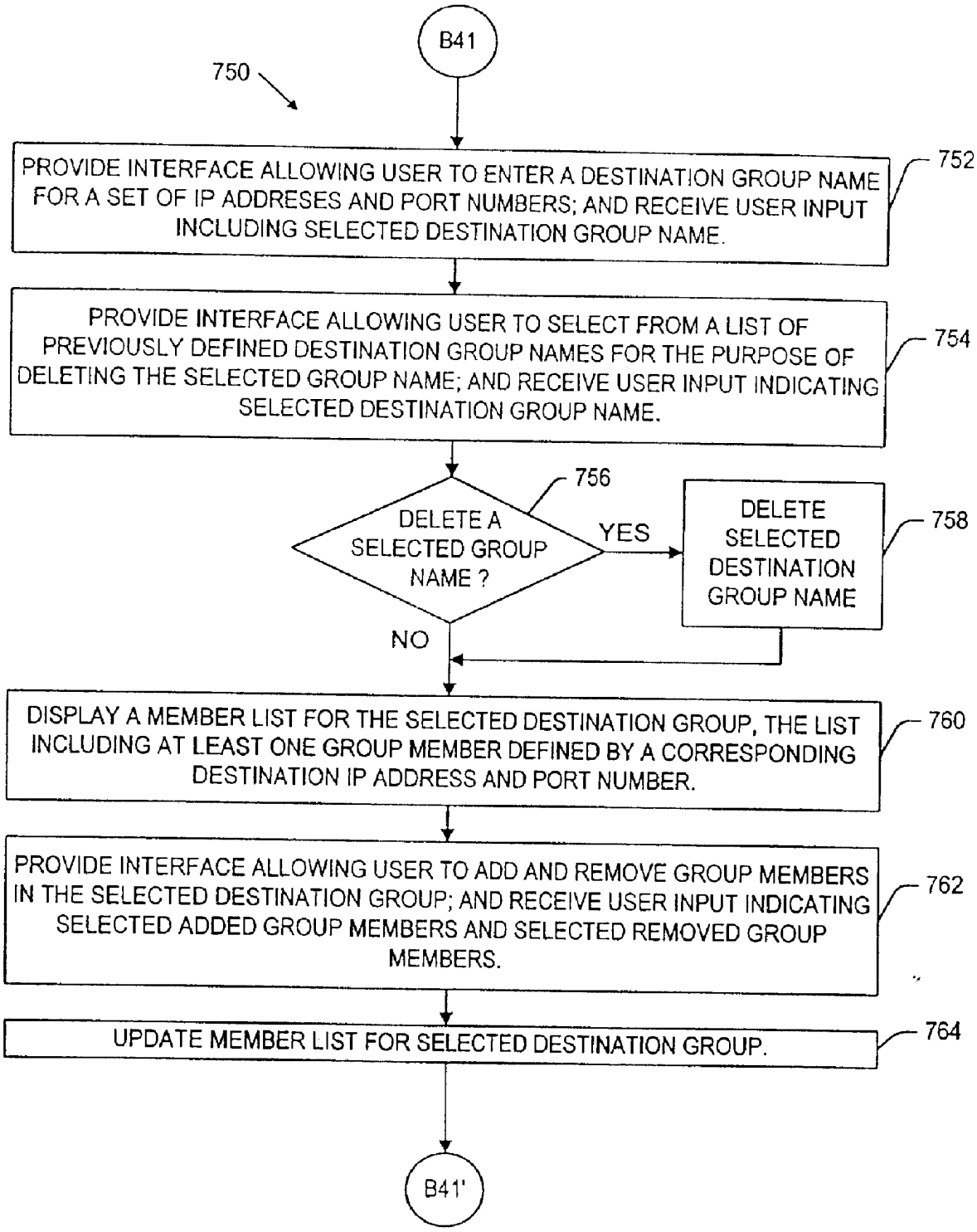

B41
750
PROVIDE INTERFACE ALLOWING USER TO ENTER A DESTINATION GROUP NAME FOR A SET OF IP ADDRESES AND PORT NUMBERS; AND RECEIVE USER INPUT INCLUDING SELECTED DESTINATION GROUP NAME.
752
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A LIST OF PREVIOUSLY DEFINED DESTINATION GROUP NAMES FOR THE PURPOSE OF DELETING THE SELECTED GROUP NAME; AND RECEIVE USER INPUT INDICATING SELECTED DESTINATION GROUP NAME.
754
756
DELETE A SELECTED GROUP NAME ?
YES
DELETE SELECTED DESTINATION GROUP NAME
758
NO
DISPLAY A MEMBER LIST FOR THE SELECTED DESTINATION GROUP, THE LIST INCLUDING AT LEAST ONE GROUP MEMBER DEFINED BY A CORRESPONDING DESTINATION IP ADDRESS AND PORT NUMBER.
760
PROVIDE INTERFACE ALLOWING USER TO ADD AND REMOVE GROUP MEMBERS IN THE SELECTED DESTINATION GROUP; AND RECEIVE USER INPUT INDICATING SELECTED ADDED GROUP MEMBERS AND SELECTED REMOVED GROUP MEMBERS.
762
UPDATE MEMBER LIST FOR SELECTED DESTINATION GROUP.
764
B41'

FIG. 20

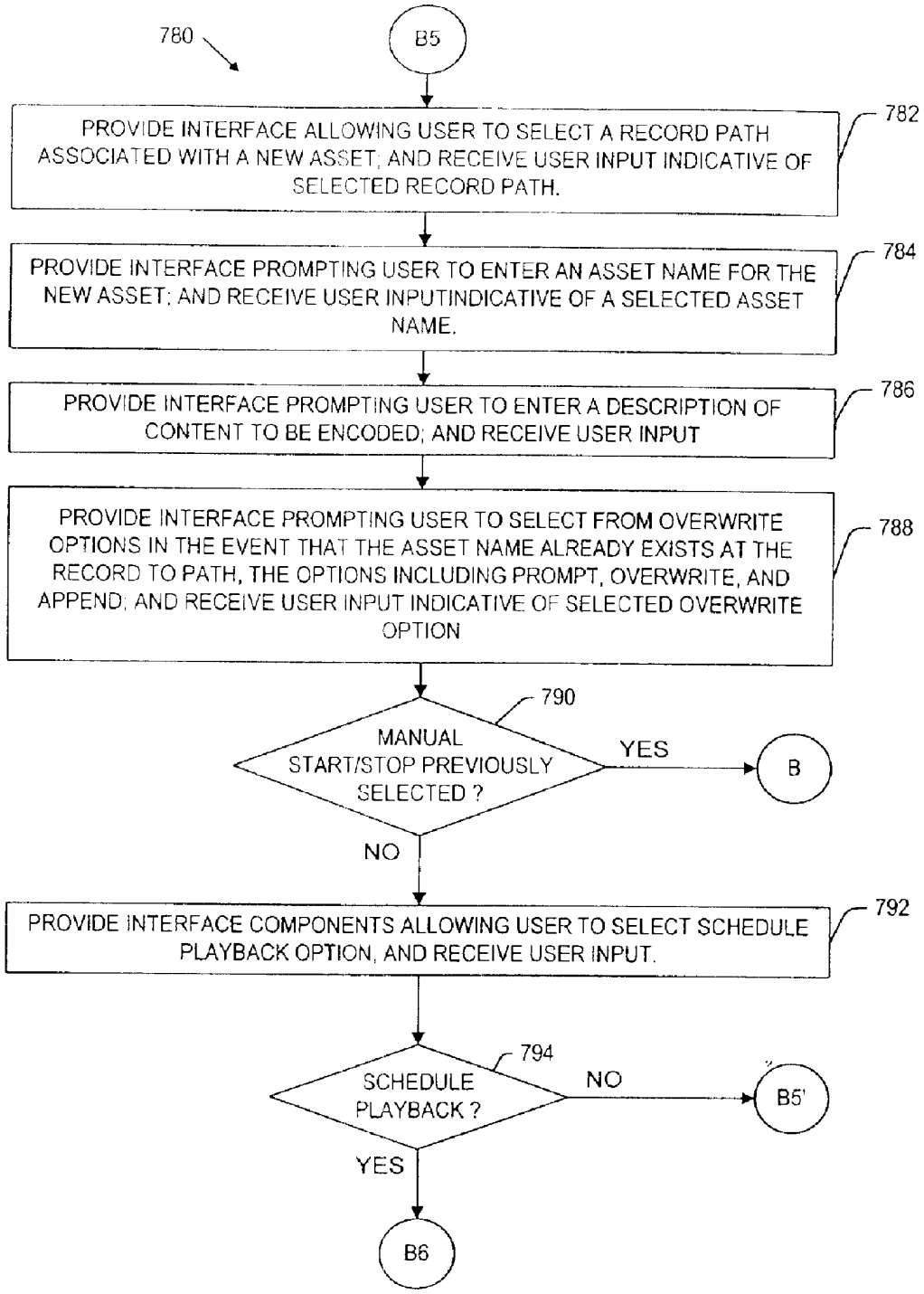

780
B5
PROVIDE INTERFACE ALLOWING USER TO SELECT A RECORD PATH ASSOCIATED WITH A NEW ASSET, AND RECEIVE USER INPUT INDICATIVE OF SELECTED RECORD PATH.
782
PROVIDE INTERFACE PROMPTING USER TO ENTER AN ASSET NAME FOR THE NEW ASSET; AND RECEIVE USER INPUTINDICATIVE OF A SELECTED ASSET NAME.
784
PROVIDE INTERFACE PROMPTING USER TO ENTER A DESCRIPTION OF CONTENT TO BE ENCODED; AND RECEIVE USER INPUT
786
PROVIDE INTERFACE PROMPTING USER TO SELECT FROM OVERWRITE OPTIONS IN THE EVENT THAT THE ASSET NAME ALREADY EXISTS AT THE RECORD TO PATH, THE OPTIONS INCLUDING PROMPT, OVERWRITE, AND APPEND, AND RECEIVE USER INPUT INDICATIVE OF SELECTED OVERWRITE OPTION
788
790
MANUAL START/STOP PREVIOUSLY SELECTED ?
YES
B
NO
PROVIDE INTERFACE COMPONENTS ALLOWING USER TO SELECT SCHEDULE PLAYBACK OPTION, AND RECEIVE USER INPUT.
792
794
SCHEDULE PLAYBACK ?
NO
B5'
YES
B6

FIG. 21

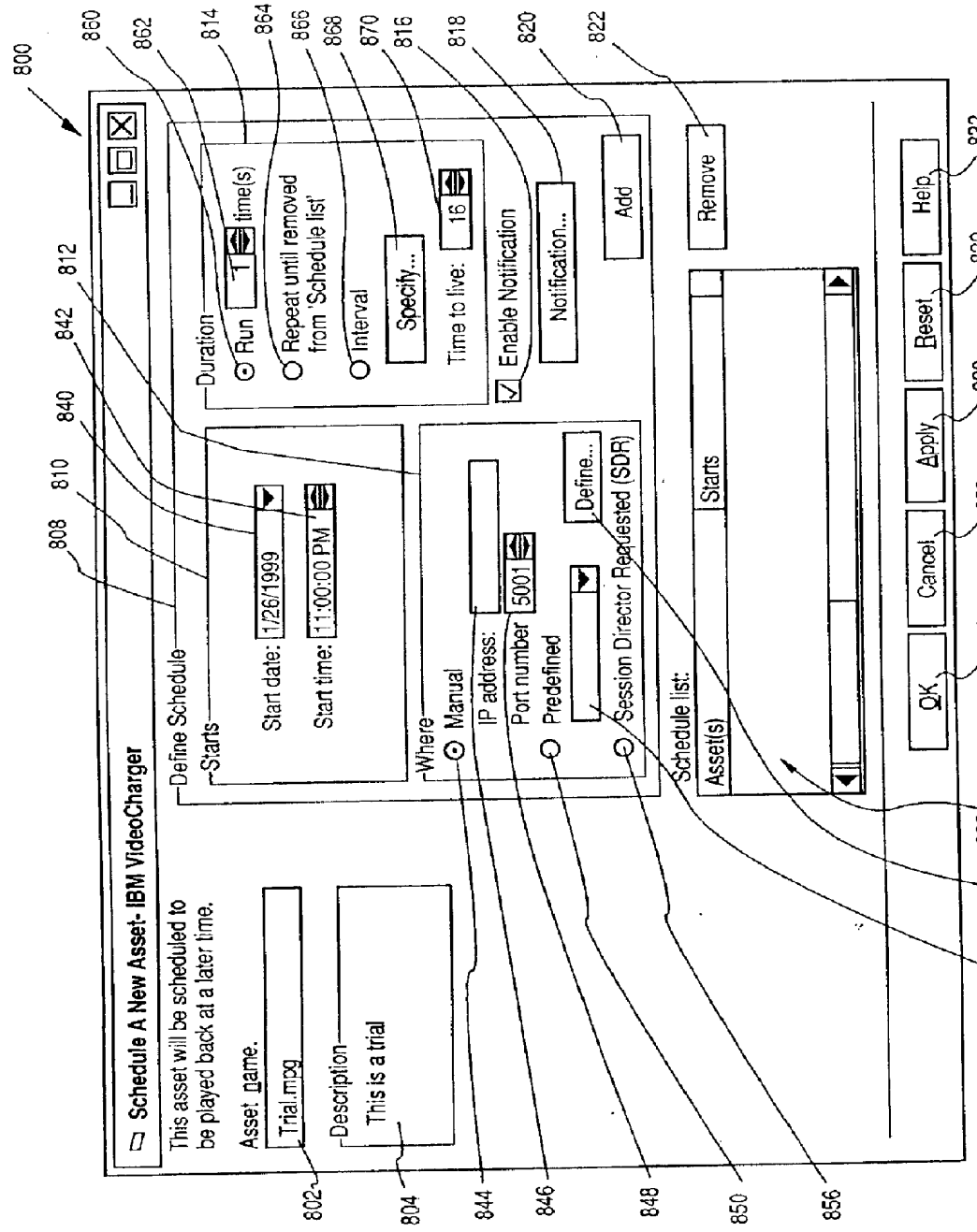

800
Schedule A New Asset- IBM VideoCharger
This asset will be scheduled to be played back at a later time.
Asset name.
Trial.mpg
Description
This is a trial
Define Schedule
Starts
Start date: 1/26/1999
Start time: 11:00:00 PM
Where
Manual
IP address:
Port number 5001
Predefined
Define...
Session Director Requested (SDR)
Duration
Run 1 time(s)
Repeat until removed from 'Schedule list'
Interval
Specify...
Time to live: 16
Enable Notification
Notification...
Add
Remove
Schedule list:
Asset(s)
Starts
OK
Cancel
Apply
Reset
Help
802
804
806
808
810
812
814
816
818
820
822
824
826
828
830
832
840
842
844
846
848
850
852
854
856
860
862
864
866
868
870

FIG. 22

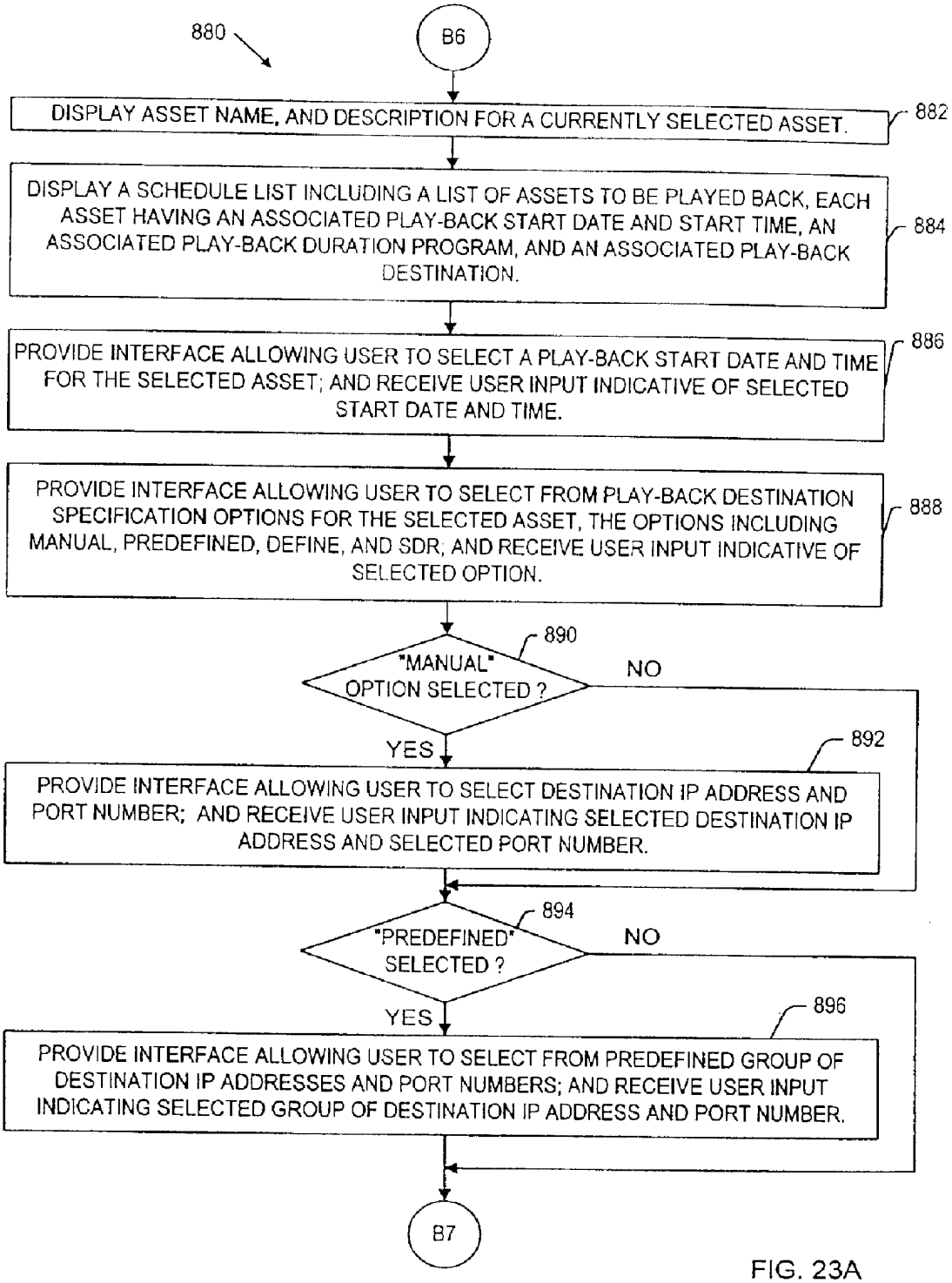

880
B6
DISPLAY ASSET NAME, AND DESCRIPTION FOR A CURRENTLY SELECTED ASSET.
882
DISPLAY A SCHEDULE LIST INCLUDING A LIST OF ASSETS TO BE PLAYED BACK, EACH ASSET HAVING AN ASSOCIATED PLAY-BACK START DATE AND START TIME, AN ASSOCIATED PLAY-BACK DURATION PROGRAM, AND AN ASSOCIATED PLAY-BACK DESTINATION.
884
PROVIDE INTERFACE ALLOWING USER TO SELECT A PLAY-BACK START DATE AND TIME FOR THE SELECTED ASSET; AND RECEIVE USER INPUT INDICATIVE OF SELECTED START DATE AND TIME.
886
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM PLAY-BACK DESTINATION SPECIFICATION OPTIONS FOR THE SELECTED ASSET, THE OPTIONS INCLUDING MANUAL, PREDEFINED, DEFINE, AND SDR; AND RECEIVE USER INPUT INDICATIVE OF SELECTED OPTION.
888
890
"MANUAL" OPTION SELECTED ?
NO
YES
892
PROVIDE INTERFACE ALLOWING USER TO SELECT DESTINATION IP ADDRESS AND PORT NUMBER; AND RECEIVE USER INPUT INDICATING SELECTED DESTINATION IP ADDRESS AND SELECTED PORT NUMBER.
894
"PREDEFINED" SELECTED ?
NO
YES
896
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM PREDEFINED GROUP OF DESTINATION IP ADDRESSES AND PORT NUMBERS; AND RECEIVE USER INPUT INDICATING SELECTED GROUP OF DESTINATION IP ADDRESS AND PORT NUMBER.
B7

FIG. 23A

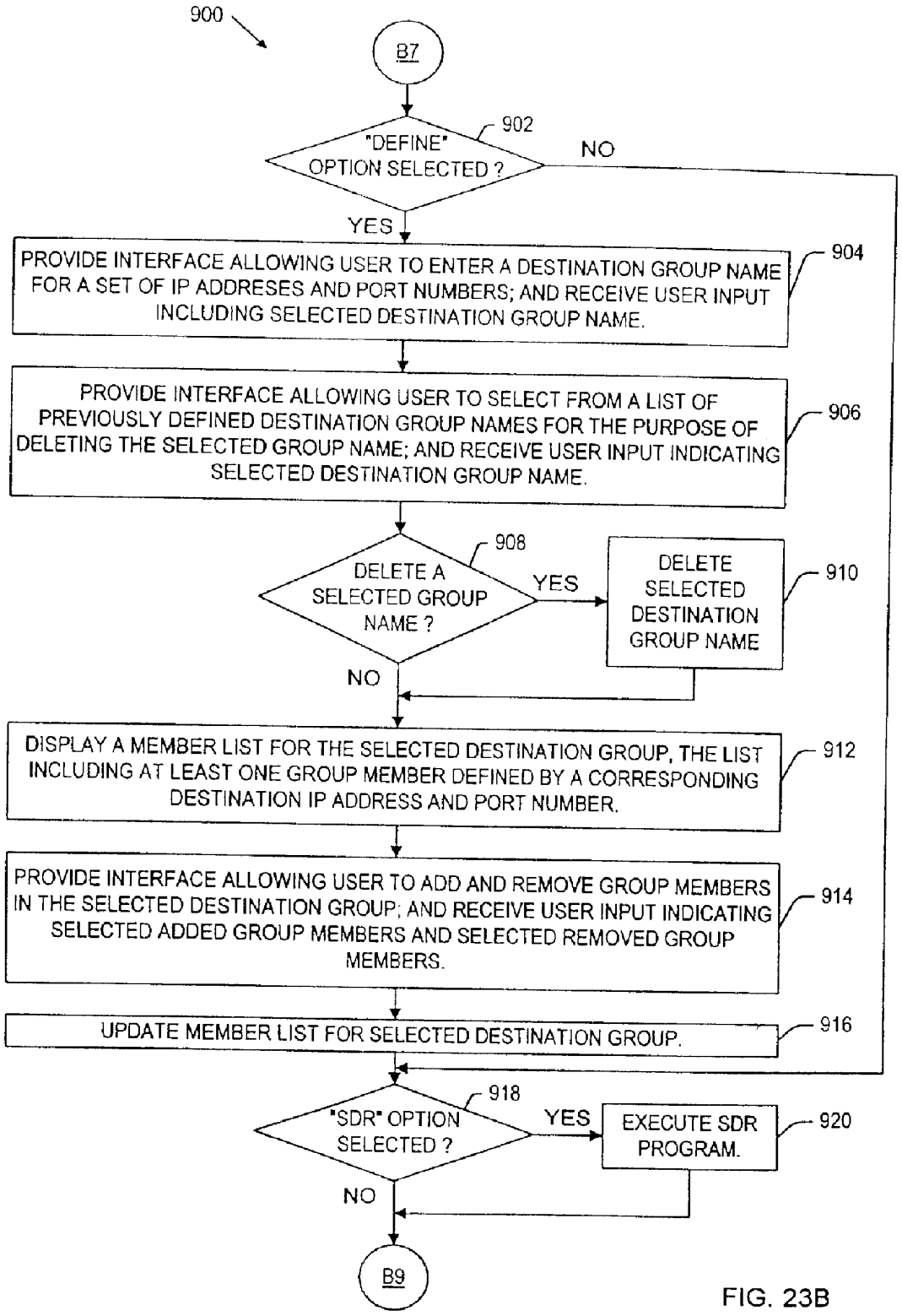

900
B7
902
"DEFINE" OPTION SELECTED ?
NO
YES
PROVIDE INTERFACE ALLOWING USER TO ENTER A DESTINATION GROUP NAME FOR A SET OF IP ADDRESES AND PORT NUMBERS; AND RECEIVE USER INPUT INCLUDING SELECTED DESTINATION GROUP NAME.
904
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A LIST OF PREVIOUSLY DEFINED DESTINATION GROUP NAMES FOR THE PURPOSE OF DELETING THE SELECTED GROUP NAME; AND RECEIVE USER INPUT INDICATING SELECTED DESTINATION GROUP NAME.
906
908
DELETE A SELECTED GROUP NAME ?
YES
DELETE SELECTED DESTINATION GROUP NAME
910
NO
DISPLAY A MEMBER LIST FOR THE SELECTED DESTINATION GROUP, THE LIST INCLUDING AT LEAST ONE GROUP MEMBER DEFINED BY A CORRESPONDING DESTINATION IP ADDRESS AND PORT NUMBER.
912
PROVIDE INTERFACE ALLOWING USER TO ADD AND REMOVE GROUP MEMBERS IN THE SELECTED DESTINATION GROUP; AND RECEIVE USER INPUT INDICATING SELECTED ADDED GROUP MEMBERS AND SELECTED REMOVED GROUP MEMBERS.
914
UPDATE MEMBER LIST FOR SELECTED DESTINATION GROUP.
916
918
"SDR" OPTION SELECTED ?
YES
EXECUTE SDR PROGRAM.
920
NO
B9

FIG. 23B

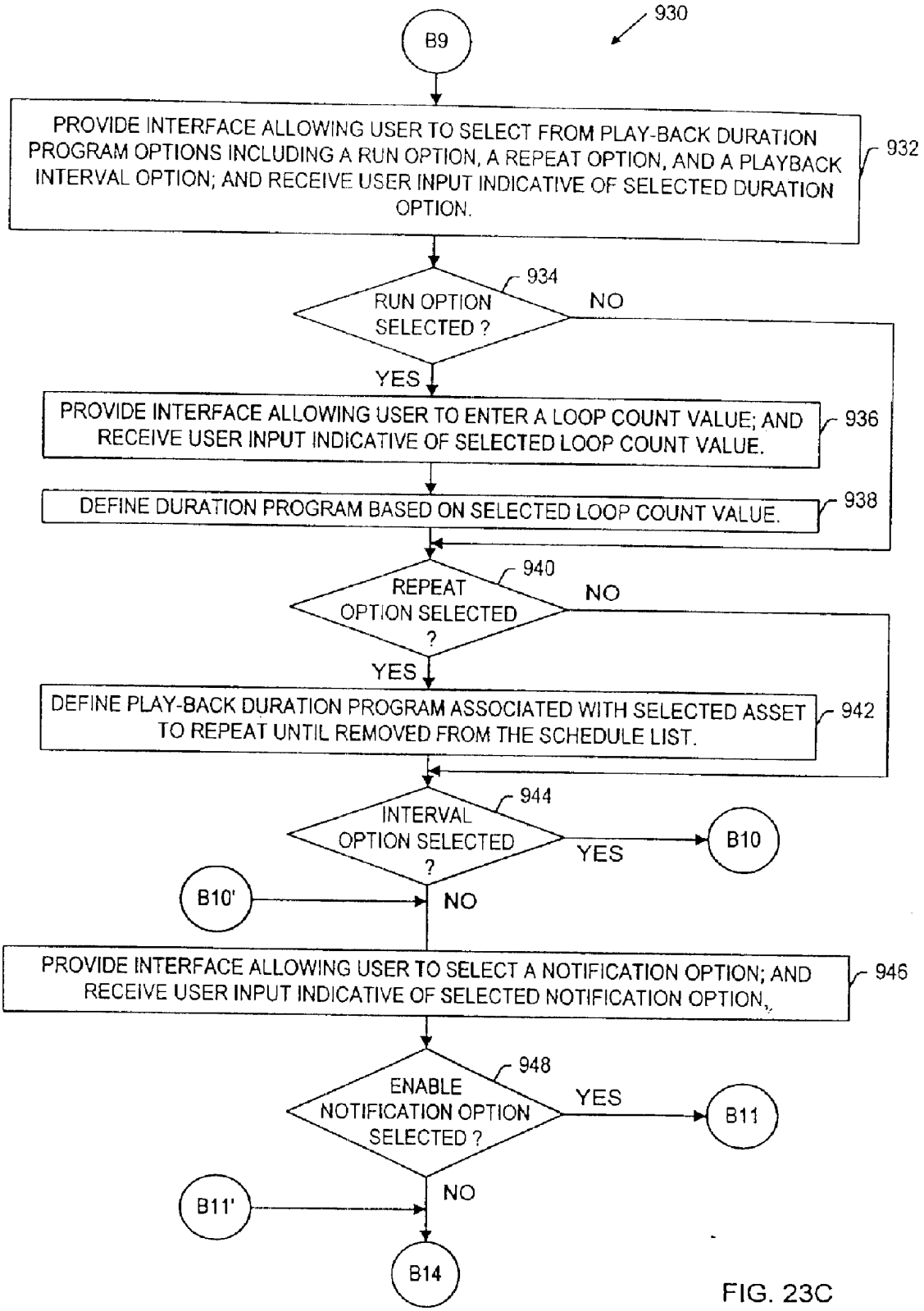

930
B9
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM PLAY-BACK DURATION PROGRAM OPTIONS INCLUDING A RUN OPTION, A REPEAT OPTION, AND A PLAYBACK INTERVAL OPTION; AND RECEIVE USER INPUT INDICATIVE OF SELECTED DURATION OPTION.
932
RUN OPTION SELECTED ?
934
NO
YES
PROVIDE INTERFACE ALLOWING USER TO ENTER A LOOP COUNT VALUE; AND RECEIVE USER INPUT INDICATIVE OF SELECTED LOOP COUNT VALUE.
936
DEFINE DURATION PROGRAM BASED ON SELECTED LOOP COUNT VALUE.
938
REPEAT OPTION SELECTED ?
940
NO
YES
DEFINE PLAY-BACK DURATION PROGRAM ASSOCIATED WITH SELECTED ASSET TO REPEAT UNTIL REMOVED FROM THE SCHEDULE LIST.
942
INTERVAL OPTION SELECTED ?
944
YES
B10
B10'
NO
PROVIDE INTERFACE ALLOWING USER TO SELECT A NOTIFICATION OPTION; AND RECEIVE USER INPUT INDICATIVE OF SELECTED NOTIFICATION OPTION,
946
ENABLE NOTIFICATION OPTION SELECTED ?
948
YES
B11
B11'
NO
B14

FIG. 23C

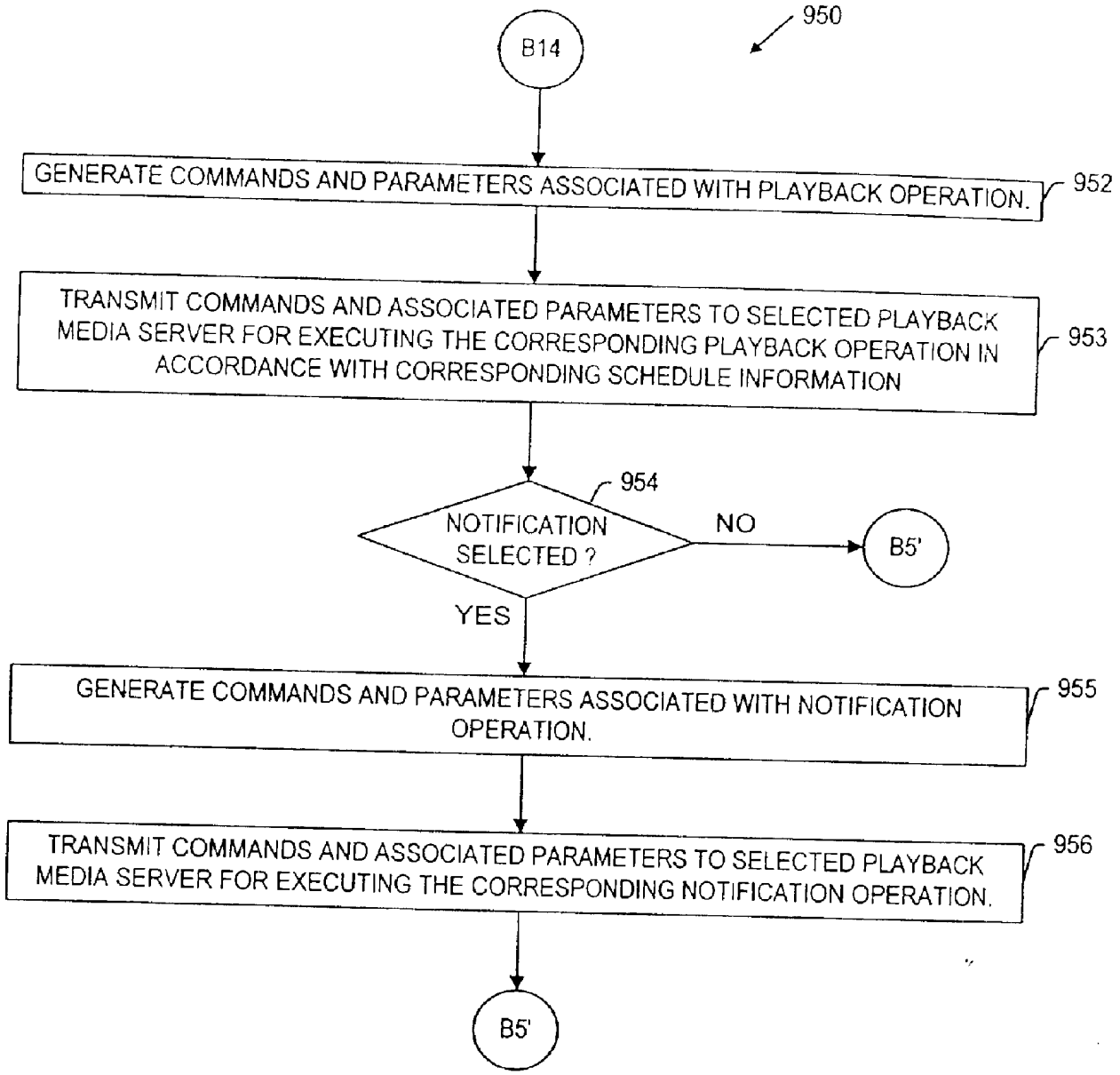

950
B14
GENERATE COMMANDS AND PARAMETERS ASSOCIATED WITH PLAYBACK OPERATION.
952
TRANSMIT COMMANDS AND ASSOCIATED PARAMETERS TO SELECTED PLAYBACK MEDIA SERVER FOR EXECUTING THE CORRESPONDING PLAYBACK OPERATION IN ACCORDANCE WITH CORRESPONDING SCHEDULE INFORMATION
953
954
NOTIFICATION SELECTED ?
NO
B5'
YES
GENERATE COMMANDS AND PARAMETERS ASSOCIATED WITH NOTIFICATION OPERATION.
955
TRANSMIT COMMANDS AND ASSOCIATED PARAMETERS TO SELECTED PLAYBACK MEDIA SERVER FOR EXECUTING THE CORRESPONDING NOTIFICATION OPERATION.
956
B5'

FIG. 23D

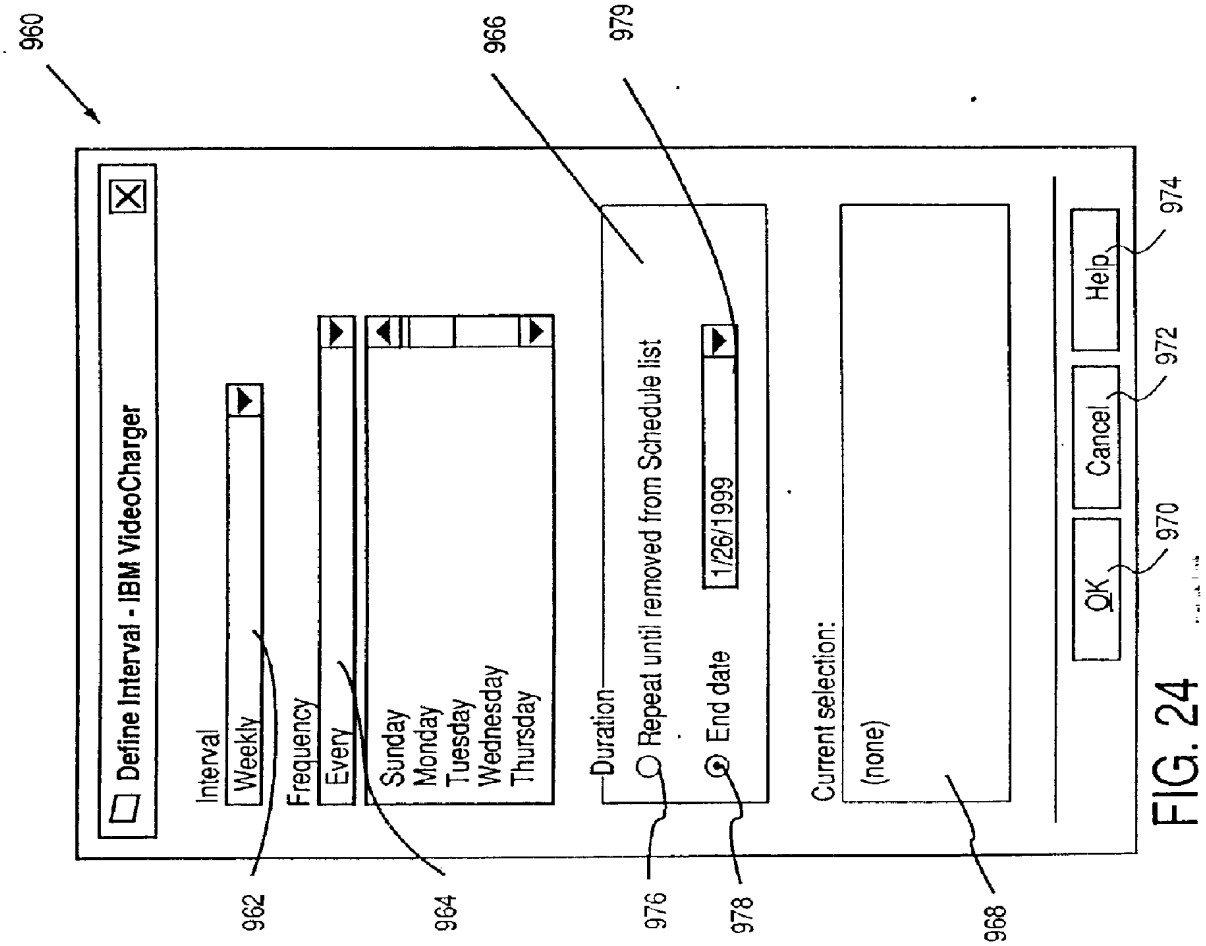

960
966
979
Define Interval - IBM VideoCharger
Interval
Weekly
Frequency
Every
Sunday
Monday
Tuesday
Wednesday
Thursday
Duration
Repeat until removed from Schedule list
End date
1/26/1999
Current selection:
(none)
OK
Cancel
Help
962
964
976
978
968
970
972
974

FIG. 24

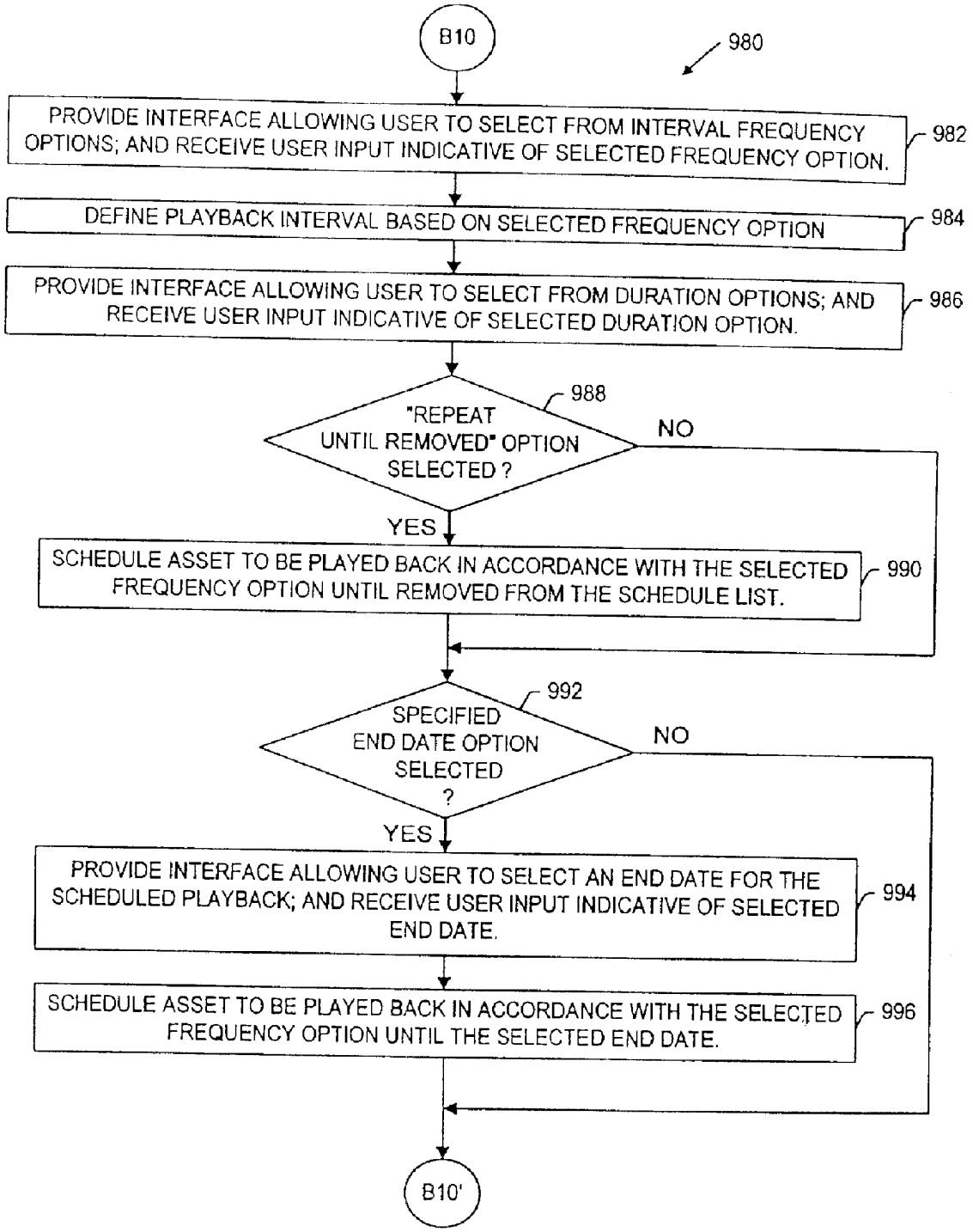

B10
980
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM INTERVAL FREQUENCY OPTIONS; AND RECEIVE USER INPUT INDICATIVE OF SELECTED FREQUENCY OPTION.
982
DEFINE PLAYBACK INTERVAL BASED ON SELECTED FREQUENCY OPTION
984
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM DURATION OPTIONS; AND RECEIVE USER INPUT INDICATIVE OF SELECTED DURATION OPTION.
986
988
"REPEAT UNTIL REMOVED" OPTION SELECTED ?
NO
YES
SCHEDULE ASSET TO BE PLAYED BACK IN ACCORDANCE WITH THE SELECTED FREQUENCY OPTION UNTIL REMOVED FROM THE SCHEDULE LIST.
990
992
SPECIFIED END DATE OPTION SELECTED ?
NO
YES
PROVIDE INTERFACE ALLOWING USER TO SELECT AN END DATE FOR THE SCHEDULED PLAYBACK; AND RECEIVE USER INPUT INDICATIVE OF SELECTED END DATE.
994
SCHEDULE ASSET TO BE PLAYED BACK IN ACCORDANCE WITH THE SELECTED FREQUENCY OPTION UNTIL THE SELECTED END DATE.
996
B10'

FIG. 25

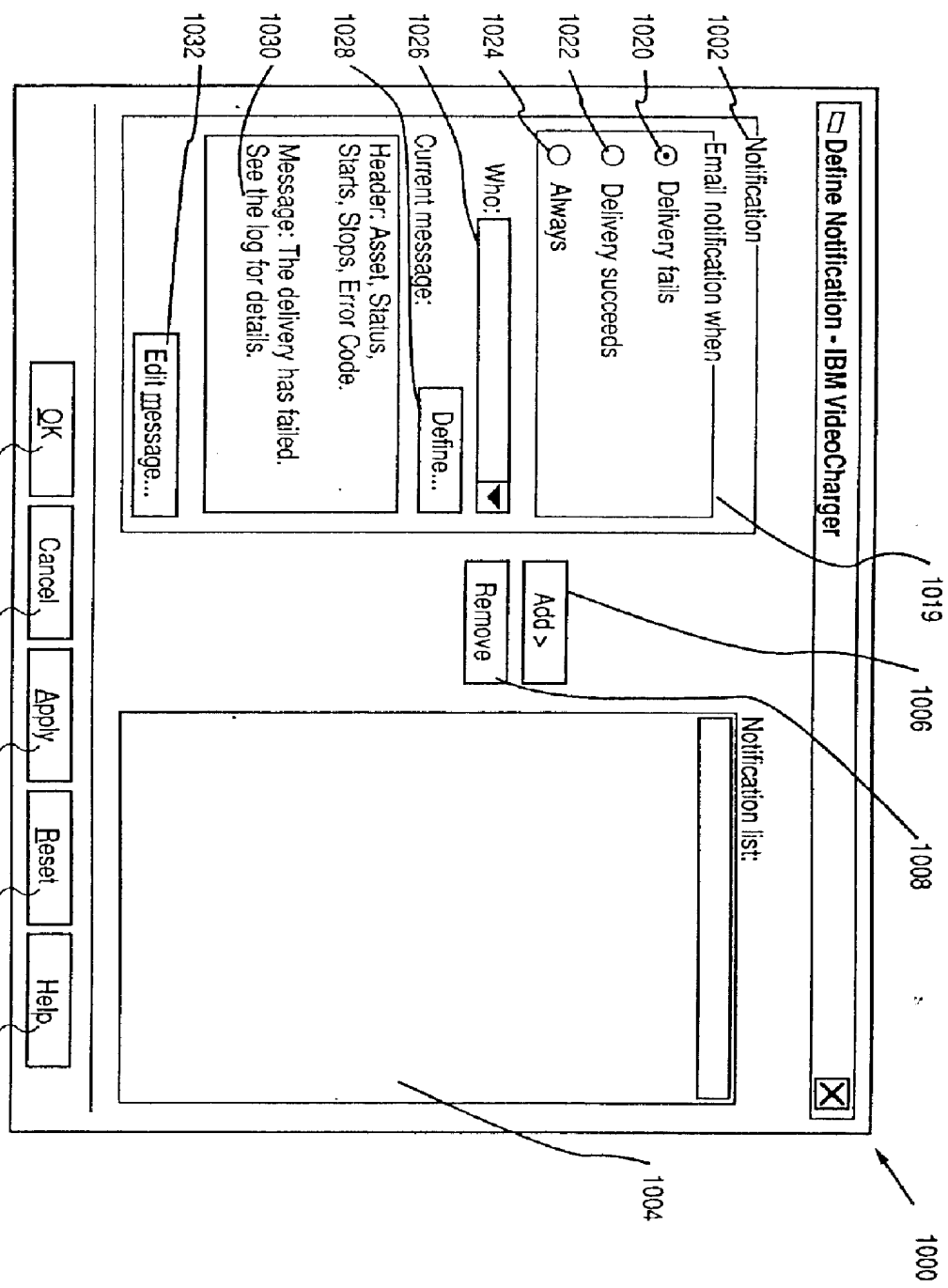

Define Notification - IBM VideoCharger
Notification
Email notification when
Delivery fails
Delivery succeeds
Always
Who:
Current message:
Define...
Header: Asset, Status, Starts, Stops, Error Code.
Message: The delivery has failed. See the log for details.
Edit message...
Add >
Remove
Notification list:
OK
Cancel
Apply
Reset
Help
1000
1002
1004
1006
1008
1010
1012
1014
1016
1018
1019
1020
1022
1024
1026
1028
1030
1032

FIG. 26

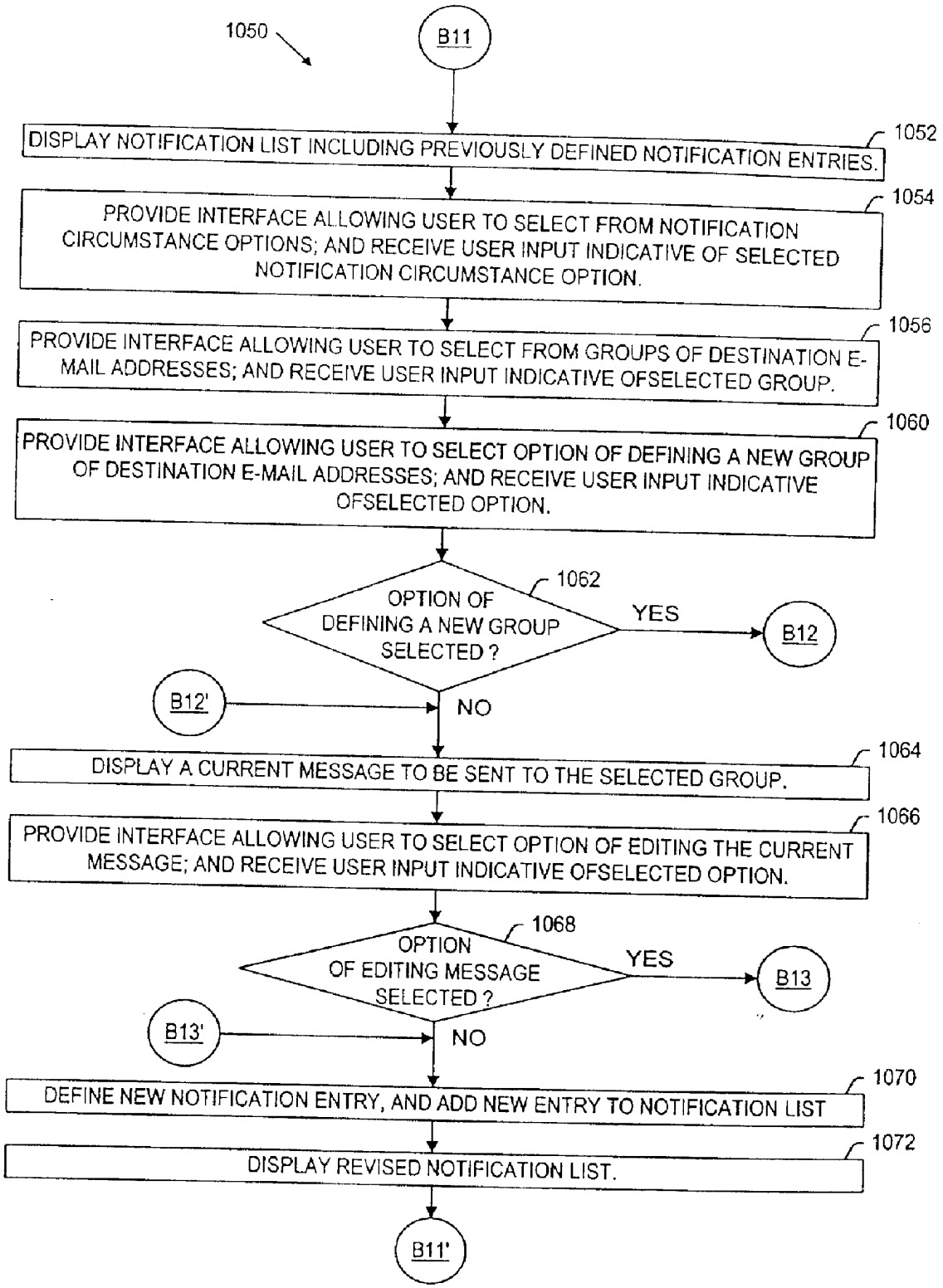

1050
B11
DISPLAY NOTIFICATION LIST INCLUDING PREVIOUSLY DEFINED NOTIFICATION ENTRIES.
1052
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM NOTIFICATION CIRCUMSTANCE OPTIONS; AND RECEIVE USER INPUT INDICATIVE OF SELECTED NOTIFICATION CIRCUMSTANCE OPTION.
1054
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM GROUPS OF DESTINATION E-MAIL ADDRESSES; AND RECEIVE USER INPUT INDICATIVE OFSELECTED GROUP.
1056
PROVIDE INTERFACE ALLOWING USER TO SELECT OPTION OF DEFINING A NEW GROUP OF DESTINATION E-MAIL ADDRESSES; AND RECEIVE USER INPUT INDICATIVE OFSELECTED OPTION.
1060
OPTION OF DEFINING A NEW GROUP SELECTED ?
1062
YES
B12
B12'
NO
DISPLAY A CURRENT MESSAGE TO BE SENT TO THE SELECTED GROUP.
1064
PROVIDE INTERFACE ALLOWING USER TO SELECT OPTION OF EDITING THE CURRENT MESSAGE; AND RECEIVE USER INPUT INDICATIVE OFSELECTED OPTION.
1066
OPTION OF EDITING MESSAGE SELECTED ?
1068
YES
B13
B13'
NO
DEFINE NEW NOTIFICATION ENTRY, AND ADD NEW ENTRY TO NOTIFICATION LIST
1070
DISPLAY REVISED NOTIFICATION LIST.
1072
B11'

FIG. 27

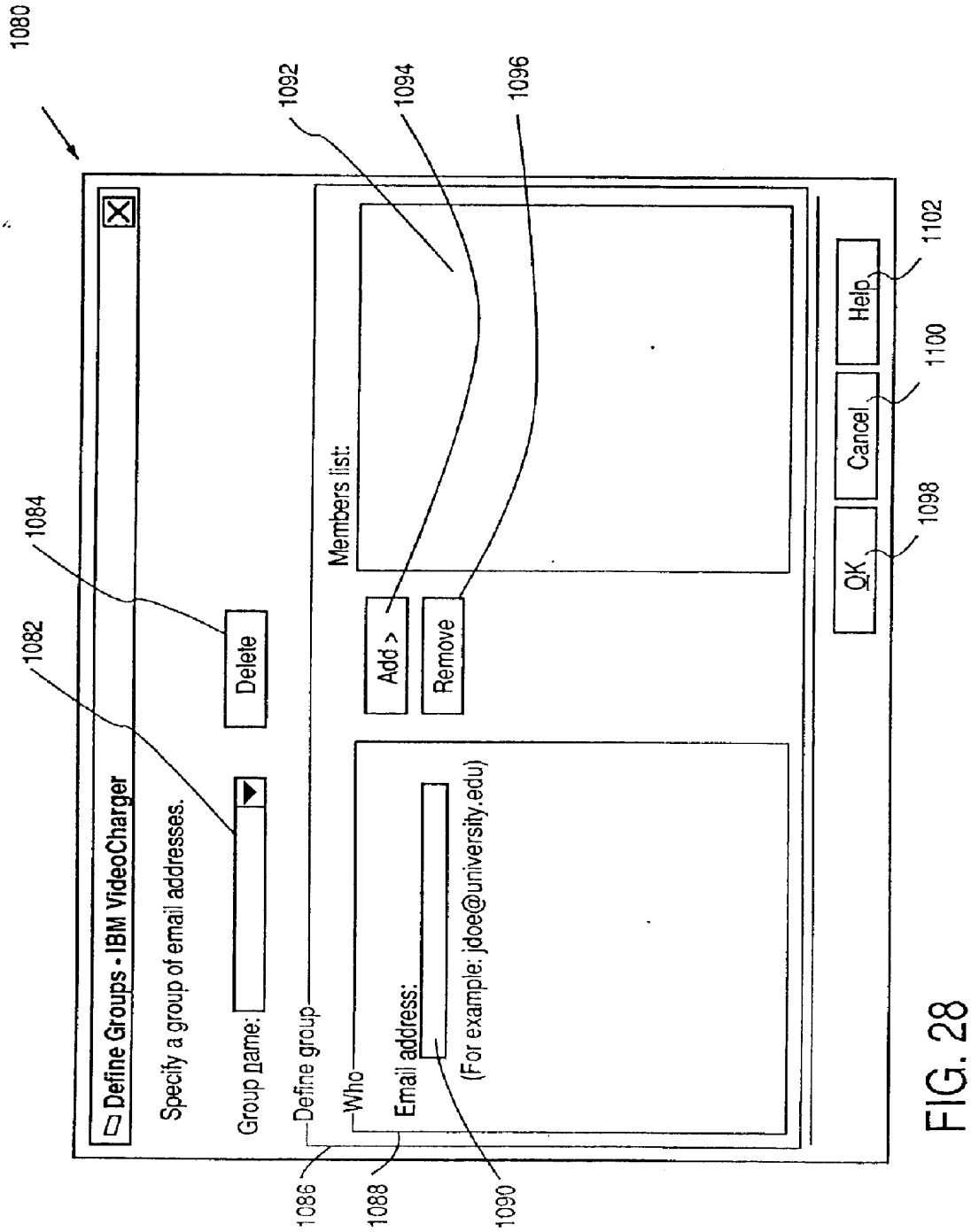

1080
1092
1094
1096
1084
1082
Define Groups - IBM VideoCharger
Specify a group of email addresses.
Group name:
Delete
Define group
Who
Email address:
(For example: jdoe@university.edu)
Members list:
Add >
Remove
OK
Cancel
Help
1102
1100
1098
1086
1088
1090

FIG. 28

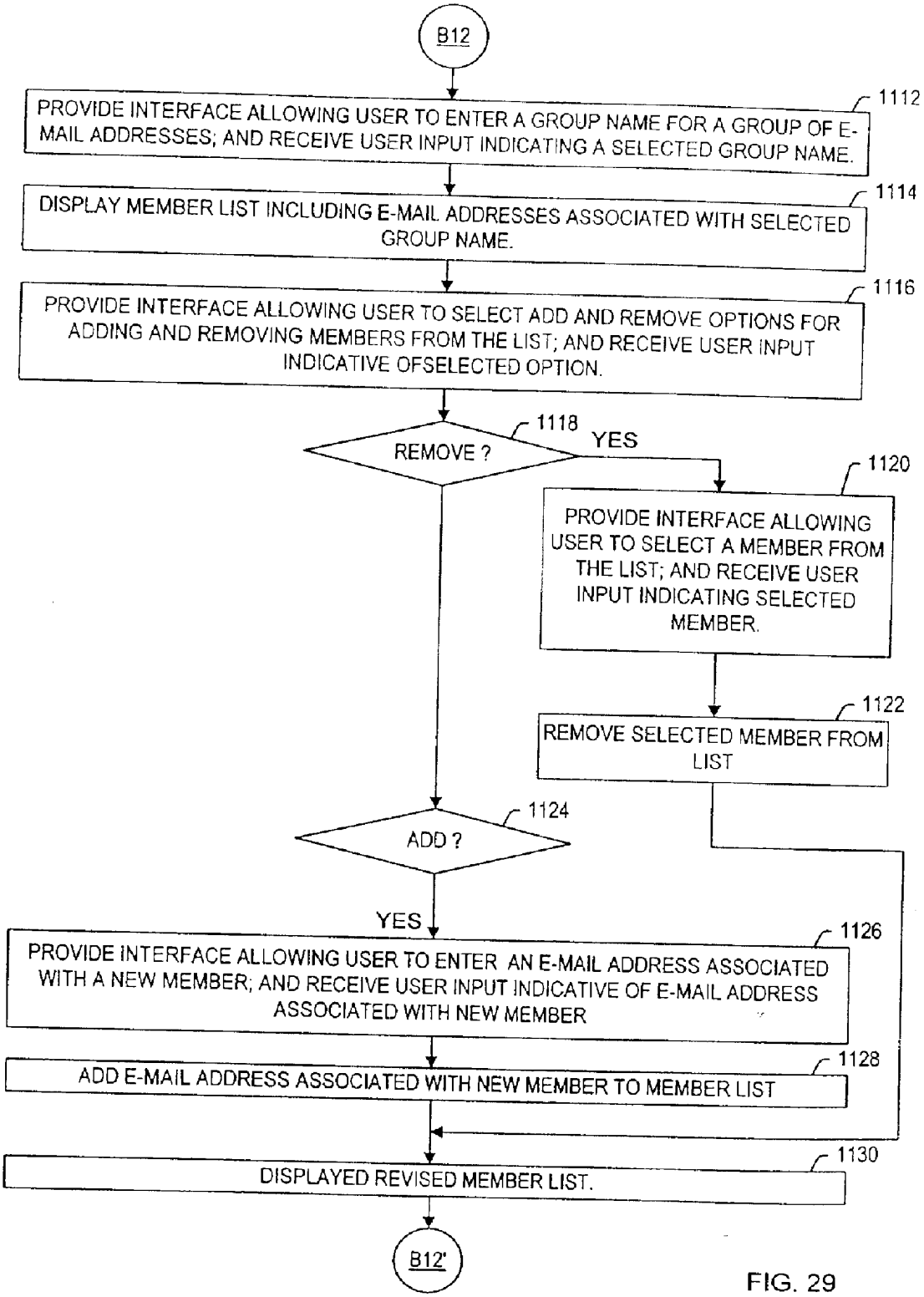

B12
PROVIDE INTERFACE ALLOWING USER TO ENTER A GROUP NAME FOR A GROUP OF E-MAIL ADDRESSES; AND RECEIVE USER INPUT INDICATING A SELECTED GROUP NAME.
1112
DISPLAY MEMBER LIST INCLUDING E-MAIL ADDRESSES ASSOCIATED WITH SELECTED GROUP NAME.
1114
PROVIDE INTERFACE ALLOWING USER TO SELECT ADD AND REMOVE OPTIONS FOR ADDING AND REMOVING MEMBERS FROM THE LIST; AND RECEIVE USER INPUT INDICATIVE OFSELECTED OPTION.
1116
REMOVE ?
1118
YES
PROVIDE INTERFACE ALLOWING USER TO SELECT A MEMBER FROM THE LIST; AND RECEIVE USER INPUT INDICATING SELECTED MEMBER.
1120
REMOVE SELECTED MEMBER FROM LIST
1122
ADD ?
1124
YES
PROVIDE INTERFACE ALLOWING USER TO ENTER AN E-MAIL ADDRESS ASSOCIATED WITH A NEW MEMBER; AND RECEIVE USER INPUT INDICATIVE OF E-MAIL ADDRESS ASSOCIATED WITH NEW MEMBER
1126
ADD E-MAIL ADDRESS ASSOCIATED WITH NEW MEMBER TO MEMBER LIST
1128
DISPLAYED REVISED MEMBER LIST.
1130
B12'

FIG. 29

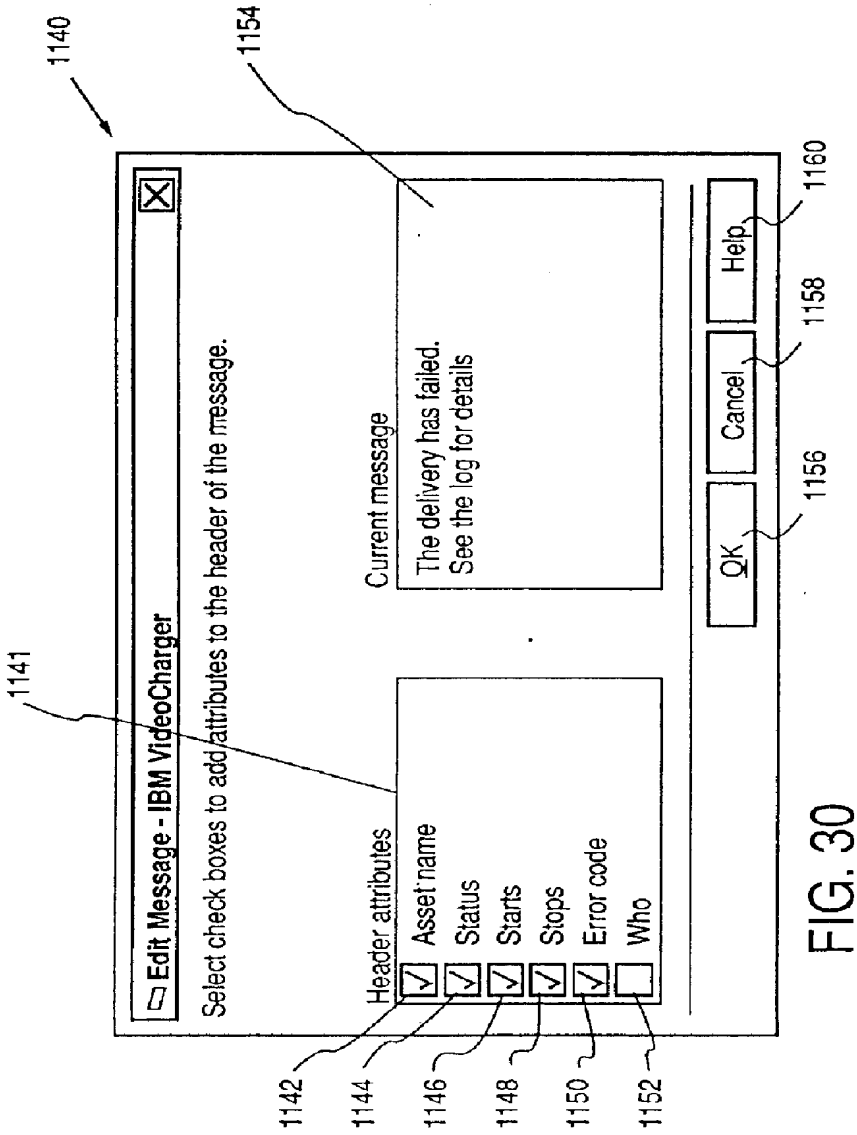

1140
1154
Edit Message - IBM VideoCharger
Select check boxes to add attributes to the header of the message.
Current message
The delivery has failed.
See the log for details
1141
Header attributes
Asset name
Status
Starts
Stops
Error code
Who
OK
Cancel
Help
1142
1144
1146
1148
1150
1152
1156
1158
1160

FIG. 30

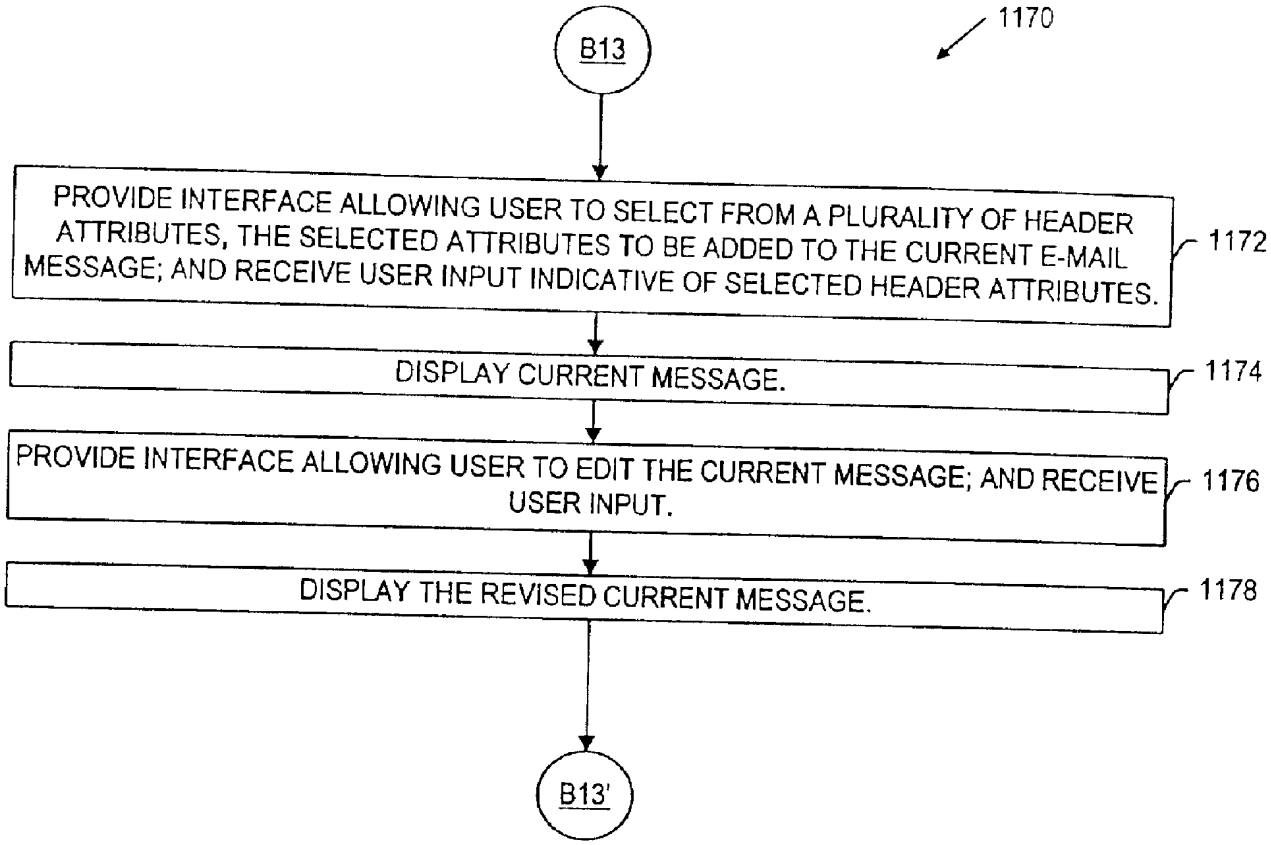

1170
B13
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A PLURALITY OF HEADER ATTRIBUTES, THE SELECTED ATTRIBUTES TO BE ADDED TO THE CURRENT E-MAIL MESSAGE; AND RECEIVE USER INPUT INDICATIVE OF SELECTED HEADER ATTRIBUTES.
1172
DISPLAY CURRENT MESSAGE.
1174
PROVIDE INTERFACE ALLOWING USER TO EDIT THE CURRENT MESSAGE; AND RECEIVE USER INPUT.
1176
DISPLAY THE REVISED CURRENT MESSAGE.
1178
B13'

FIG. 31

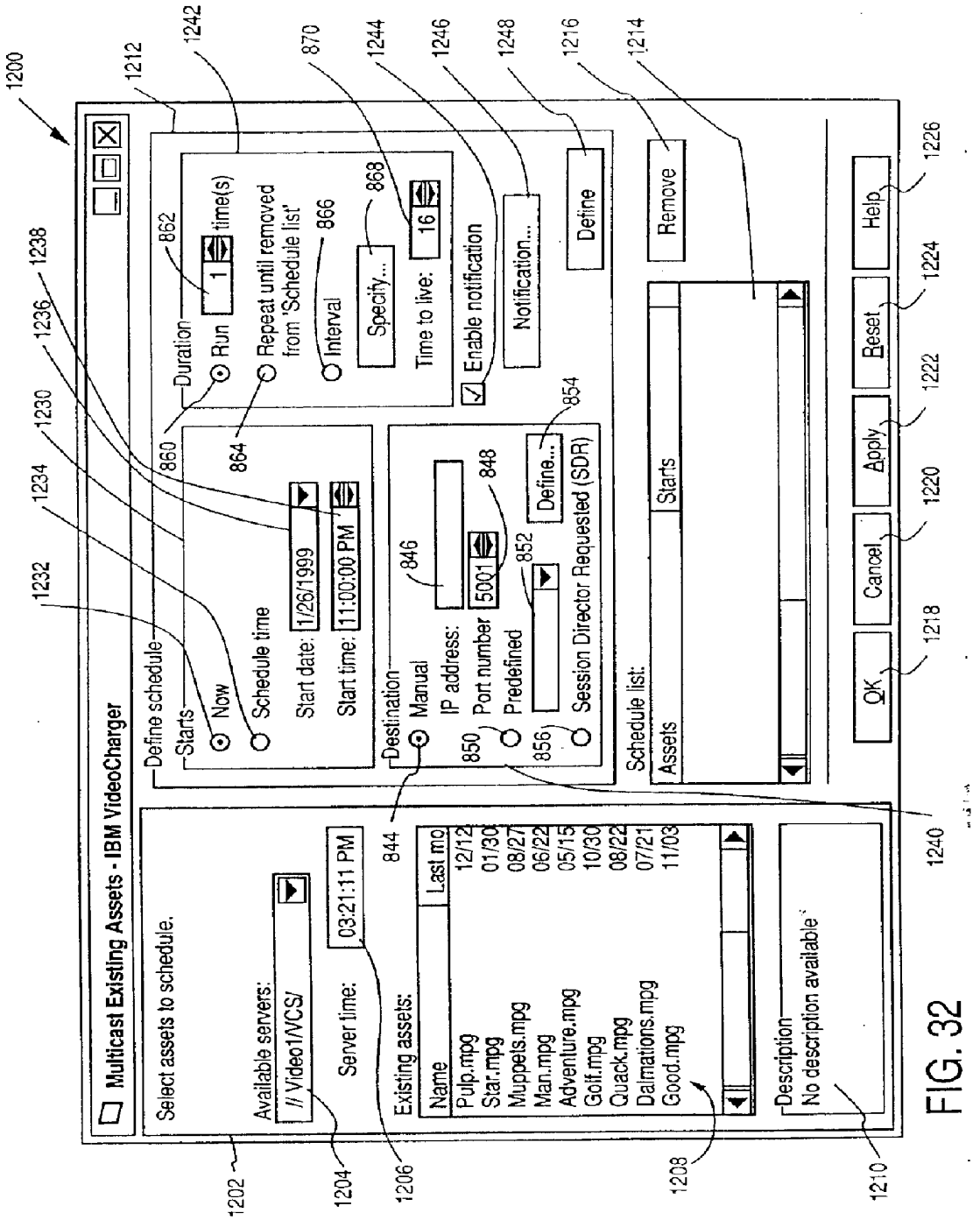

Multicast Existing Assets - IBM VideoCharger
Select assets to schedule.
Available servers:
// Video1/VCS/
Server time:
03:21:11 PM
Existing assets:
Name
Last mo
Pulp.mpg 12/12
Star.mpg 01/30
Muppets.mpg 08/27
Man.mpg 06/22
Adventure.mpg 05/15
Golf.mpg 10/30
Quack.mpg 08/22
Dalmations.mpg 07/21
Good.mpg 11/03
Description
No description available
Define schedule
Starts
Now
Schedule time
Start date: 1/26/1999
Start time: 11:00:00 PM
Duration
Run 1 time(s)
Repeat until removed from 'Schedule list'
Interval
Specify...
Time to live: 16
Enable notification
Notification...
Define
Destination
Manual
IP address:
Port number 5001
Predefined
Define...
Session Director Requested (SDR)
Schedule list:
Assets
Starts
Remove
OK
Cancel
Apply
Reset
Help
1200
1202
1204
1206
1208
1210
1212
1214
1216
1218
1220
1222
1224
1226
1230
1232
1234
1236
1238
1240
1242
1244
1246
1248
844
846
848
850
852
854
856
860
862
864
866
868
870

FIG. 32

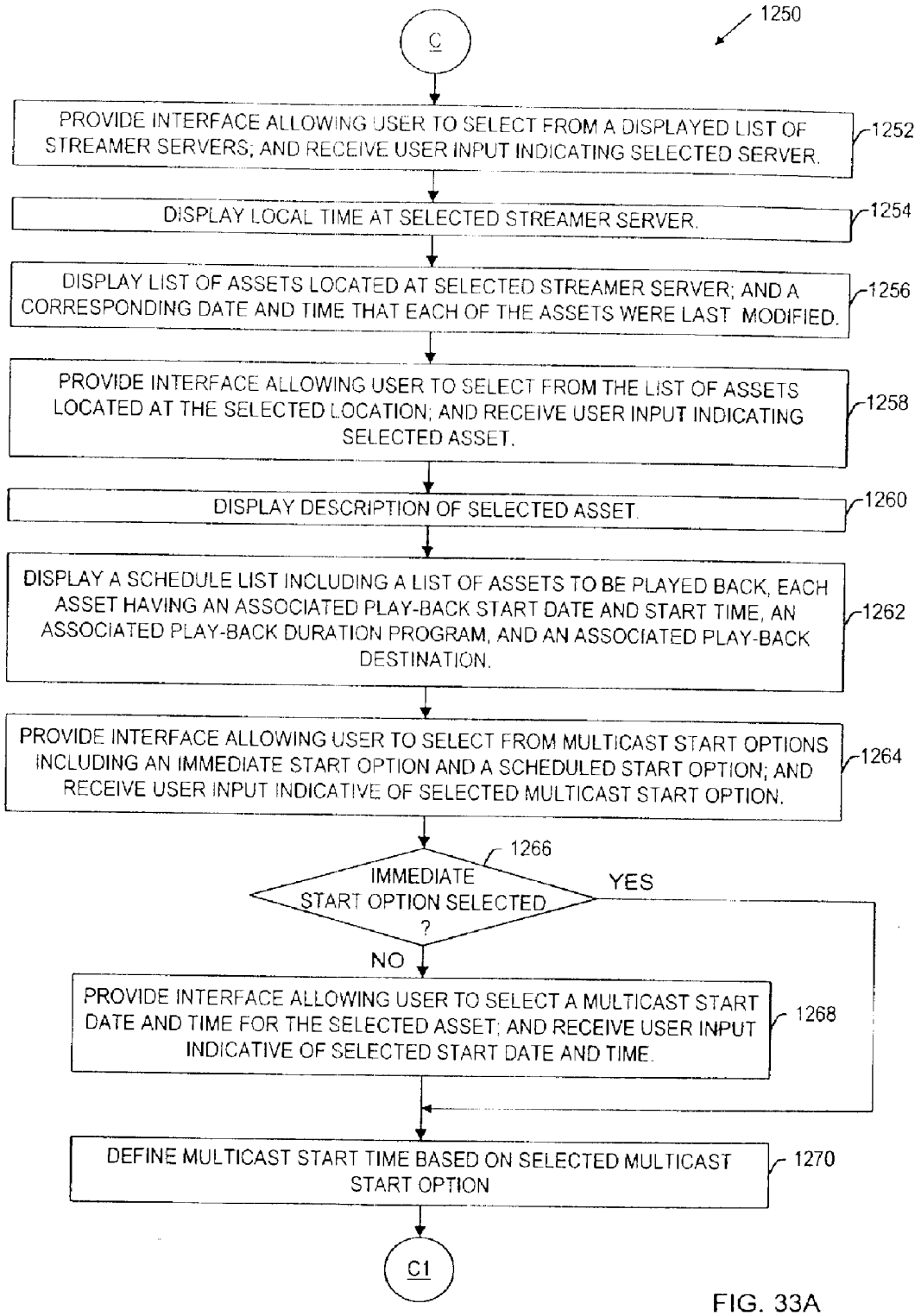

1250
C
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A DISPLAYED LIST OF STREAMER SERVERS; AND RECEIVE USER INPUT INDICATING SELECTED SERVER.
1252
DISPLAY LOCAL TIME AT SELECTED STREAMER SERVER.
1254
DISPLAY LIST OF ASSETS LOCATED AT SELECTED STREAMER SERVER; AND A CORRESPONDING DATE AND TIME THAT EACH OF THE ASSETS WERE LAST MODIFIED.
1256
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM THE LIST OF ASSETS LOCATED AT THE SELECTED LOCATION; AND RECEIVE USER INPUT INDICATING SELECTED ASSET.
1258
DISPLAY DESCRIPTION OF SELECTED ASSET.
1260
DISPLAY A SCHEDULE LIST INCLUDING A LIST OF ASSETS TO BE PLAYED BACK, EACH ASSET HAVING AN ASSOCIATED PLAY-BACK START DATE AND START TIME, AN ASSOCIATED PLAY-BACK DURATION PROGRAM, AND AN ASSOCIATED PLAY-BACK DESTINATION.
1262
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM MULTICAST START OPTIONS INCLUDING AN IMMEDIATE START OPTION AND A SCHEDULED START OPTION; AND RECEIVE USER INPUT INDICATIVE OF SELECTED MULTICAST START OPTION.
1264
1266
IMMEDIATE START OPTION SELECTED ?
YES
NO
PROVIDE INTERFACE ALLOWING USER TO SELECT A MULTICAST START DATE AND TIME FOR THE SELECTED ASSET; AND RECEIVE USER INPUT INDICATIVE OF SELECTED START DATE AND TIME.
1268
DEFINE MULTICAST START TIME BASED ON SELECTED MULTICAST START OPTION
1270
C1

FIG. 33A

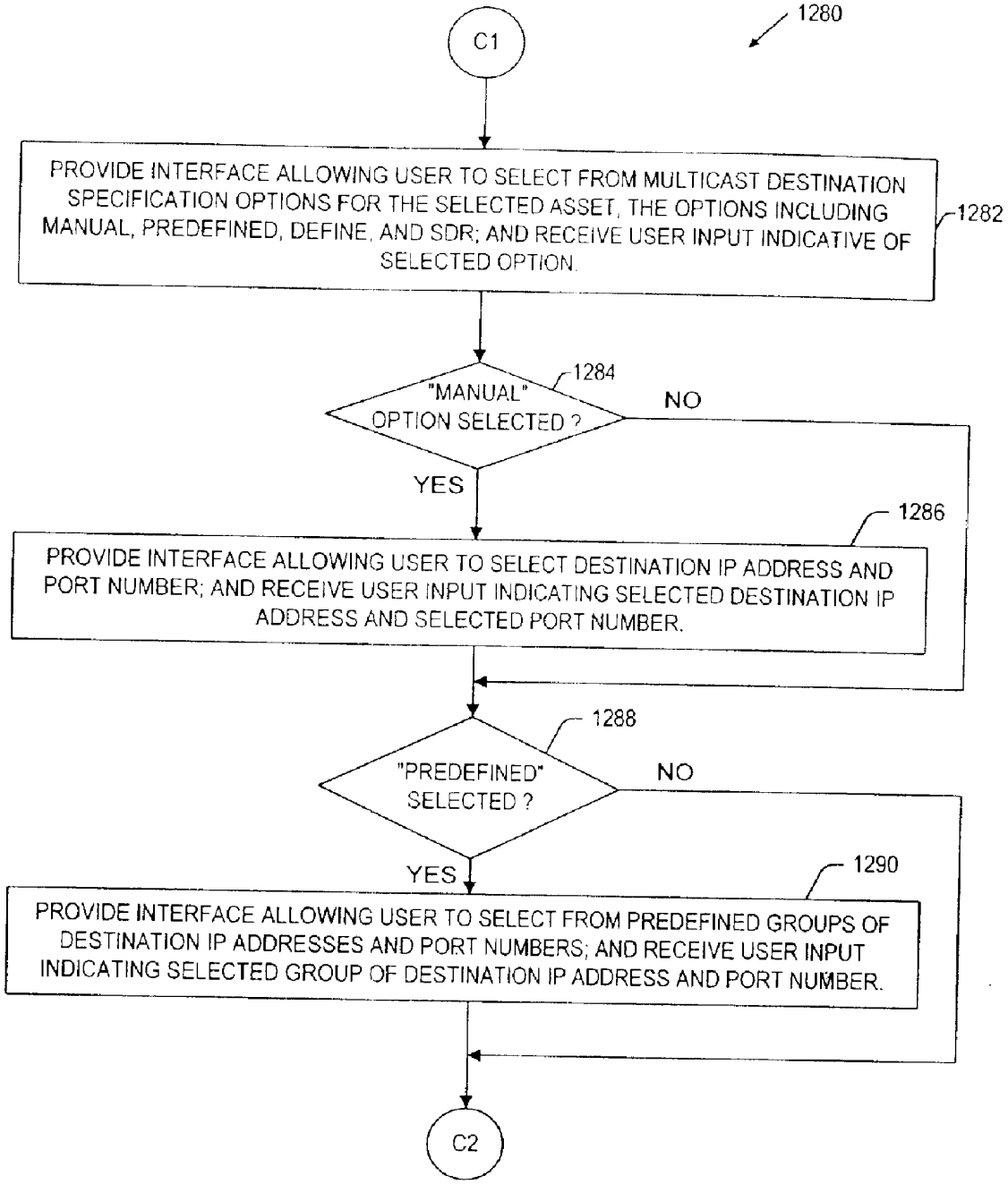

1280
C1
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM MULTICAST DESTINATION SPECIFICATION OPTIONS FOR THE SELECTED ASSET, THE OPTIONS INCLUDING MANUAL, PREDEFINED, DEFINE, AND SDR; AND RECEIVE USER INPUT INDICATIVE OF SELECTED OPTION.
1282
"MANUAL" OPTION SELECTED ?
1284
NO
YES
PROVIDE INTERFACE ALLOWING USER TO SELECT DESTINATION IP ADDRESS AND PORT NUMBER; AND RECEIVE USER INPUT INDICATING SELECTED DESTINATION IP ADDRESS AND SELECTED PORT NUMBER.
1286
"PREDEFINED" SELECTED ?
1288
NO
YES
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM PREDEFINED GROUPS OF DESTINATION IP ADDRESSES AND PORT NUMBERS; AND RECEIVE USER INPUT INDICATING SELECTED GROUP OF DESTINATION IP ADDRESS AND PORT NUMBER.
1290
C2

FIG. 33B

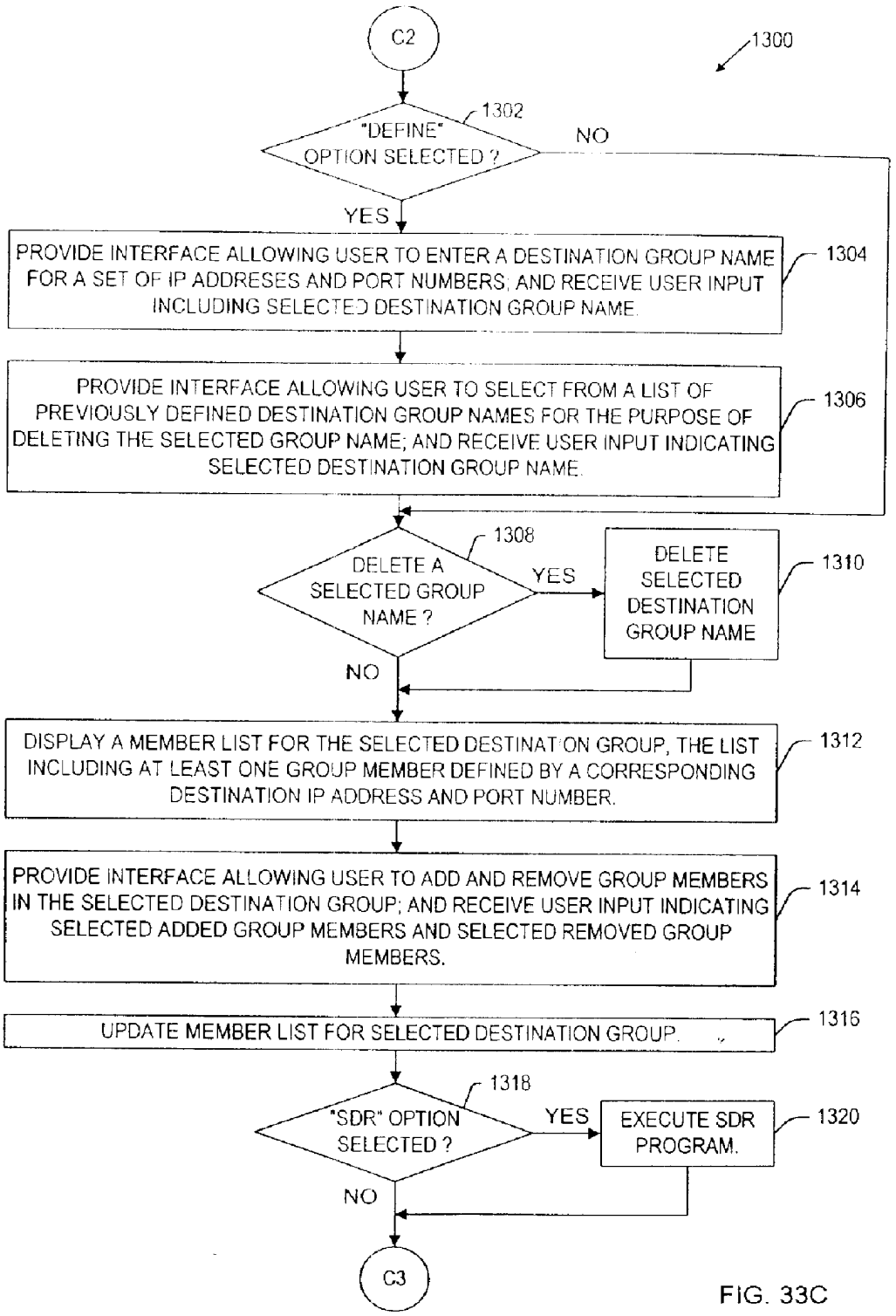

C2
1300
1302
"DEFINE" OPTION SELECTED ?
NO
YES
PROVIDE INTERFACE ALLOWING USER TO ENTER A DESTINATION GROUP NAME FOR A SET OF IP ADDRESES AND PORT NUMBERS; AND RECEIVE USER INPUT INCLUDING SELECTED DESTINATION GROUP NAME.
1304
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM A LIST OF PREVIOUSLY DEFINED DESTINATION GROUP NAMES FOR THE PURPOSE OF DELETING THE SELECTED GROUP NAME; AND RECEIVE USER INPUT INDICATING SELECTED DESTINATION GROUP NAME.
1306
1308
DELETE A SELECTED GROUP NAME ?
YES
DELETE SELECTED DESTINATION GROUP NAME
1310
NO
DISPLAY A MEMBER LIST FOR THE SELECTED DESTINATION GROUP, THE LIST INCLUDING AT LEAST ONE GROUP MEMBER DEFINED BY A CORRESPONDING DESTINATION IP ADDRESS AND PORT NUMBER.
1312
PROVIDE INTERFACE ALLOWING USER TO ADD AND REMOVE GROUP MEMBERS IN THE SELECTED DESTINATION GROUP; AND RECEIVE USER INPUT INDICATING SELECTED ADDED GROUP MEMBERS AND SELECTED REMOVED GROUP MEMBERS.
1314
UPDATE MEMBER LIST FOR SELECTED DESTINATION GROUP.
1316
1318
"SDR" OPTION SELECTED ?
YES
EXECUTE SDR PROGRAM.
1320
NO
C3

FIG. 33C

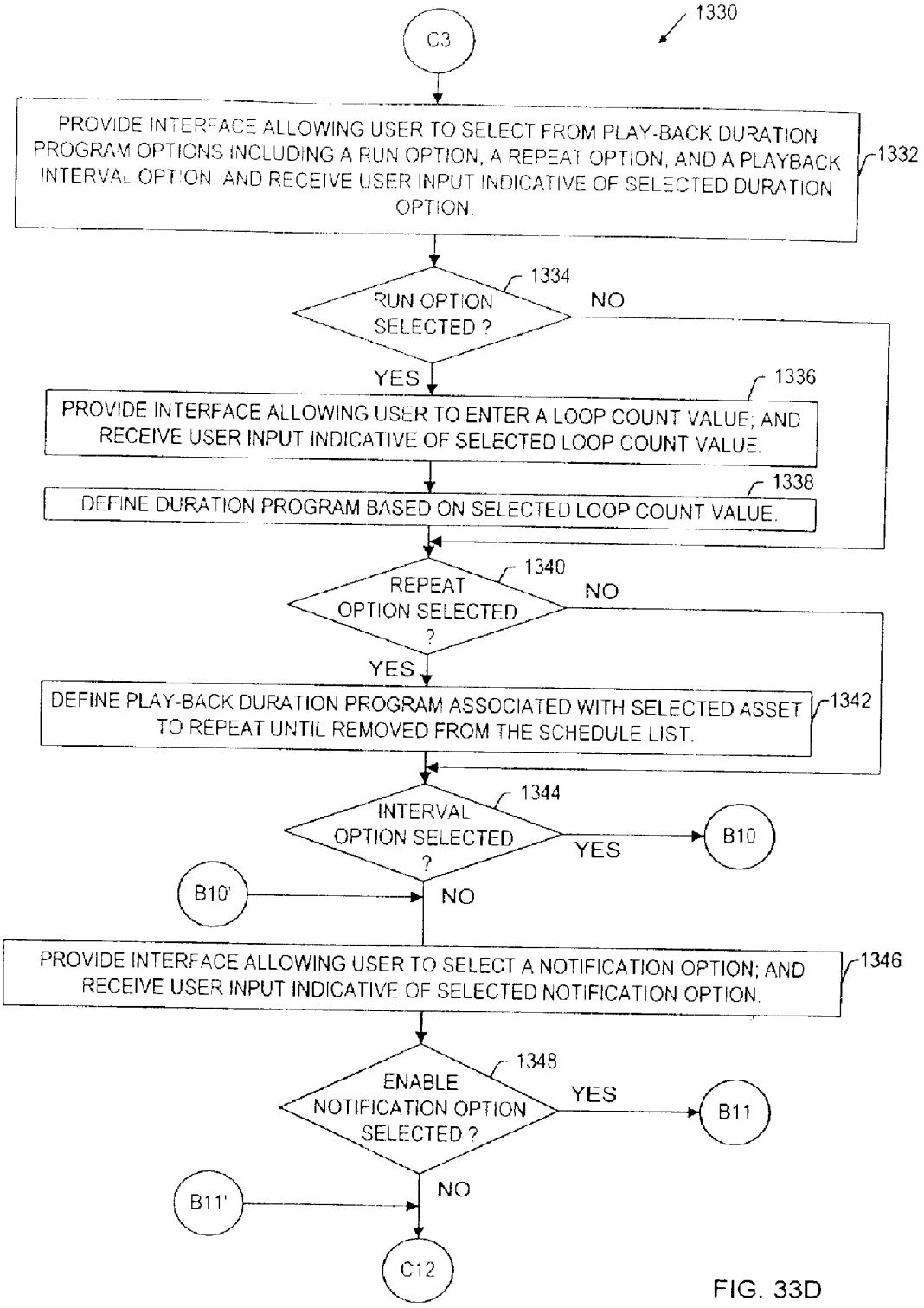

1330
C3
PROVIDE INTERFACE ALLOWING USER TO SELECT FROM PLAY-BACK DURATION PROGRAM OPTIONS INCLUDING A RUN OPTION, A REPEAT OPTION, AND A PLAYBACK INTERVAL OPTION, AND RECEIVE USER INPUT INDICATIVE OF SELECTED DURATION OPTION.
1332
1334
RUN OPTION SELECTED ?
NO
YES
1336
PROVIDE INTERFACE ALLOWING USER TO ENTER A LOOP COUNT VALUE; AND RECEIVE USER INPUT INDICATIVE OF SELECTED LOOP COUNT VALUE.
1338
DEFINE DURATION PROGRAM BASED ON SELECTED LOOP COUNT VALUE.
1340
REPEAT OPTION SELECTED ?
NO
YES
DEFINE PLAY-BACK DURATION PROGRAM ASSOCIATED WITH SELECTED ASSET TO REPEAT UNTIL REMOVED FROM THE SCHEDULE LIST.
1342
1344
INTERVAL OPTION SELECTED ?
YES
B10
B10'
NO
PROVIDE INTERFACE ALLOWING USER TO SELECT A NOTIFICATION OPTION; AND RECEIVE USER INPUT INDICATIVE OF SELECTED NOTIFICATION OPTION.
1346
1348
ENABLE NOTIFICATION OPTION SELECTED ?
YES
B11
B11'
NO
C12

FIG. 33D

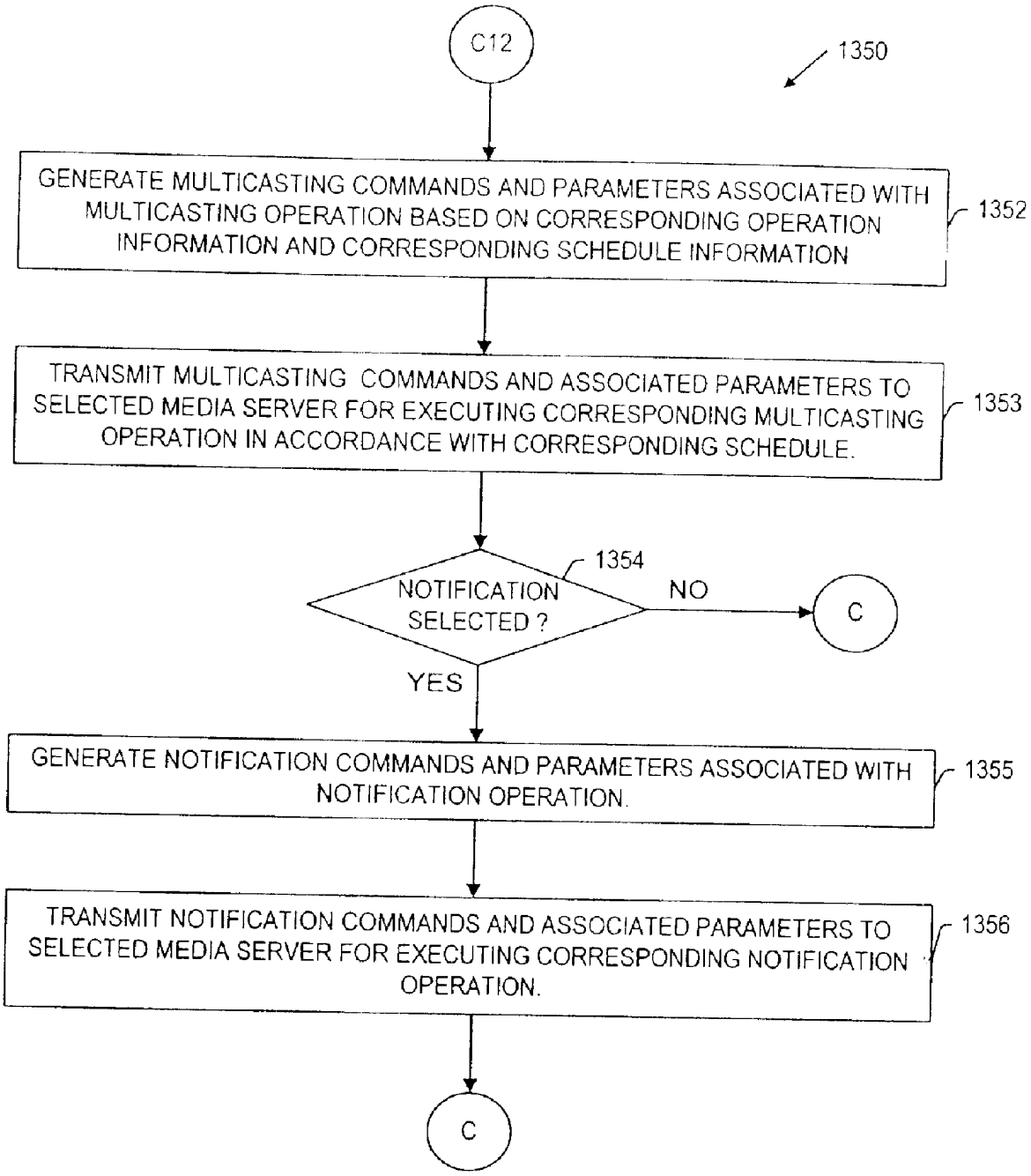

C12
1350
GENERATE MULTICASTING COMMANDS AND PARAMETERS ASSOCIATED WITH MULTICASTING OPERATION BASED ON CORRESPONDING OPERATION INFORMATION AND CORRESPONDING SCHEDULE INFORMATION
1352
TRANSMIT MULTICASTING COMMANDS AND ASSOCIATED PARAMETERS TO SELECTED MEDIA SERVER FOR EXECUTING CORRESPONDING MULTICASTING OPERATION IN ACCORDANCE WITH CORRESPONDING SCHEDULE.
1353
1354
NOTIFICATION SELECTED ?
NO
C
YES
GENERATE NOTIFICATION COMMANDS AND PARAMETERS ASSOCIATED WITH NOTIFICATION OPERATION.
1355
TRANSMIT NOTIFICATION COMMANDS AND ASSOCIATED PARAMETERS TO SELECTED MEDIA SERVER FOR EXECUTING CORRESPONDING NOTIFICATION OPERATION.
1356
C

FIG. 33E

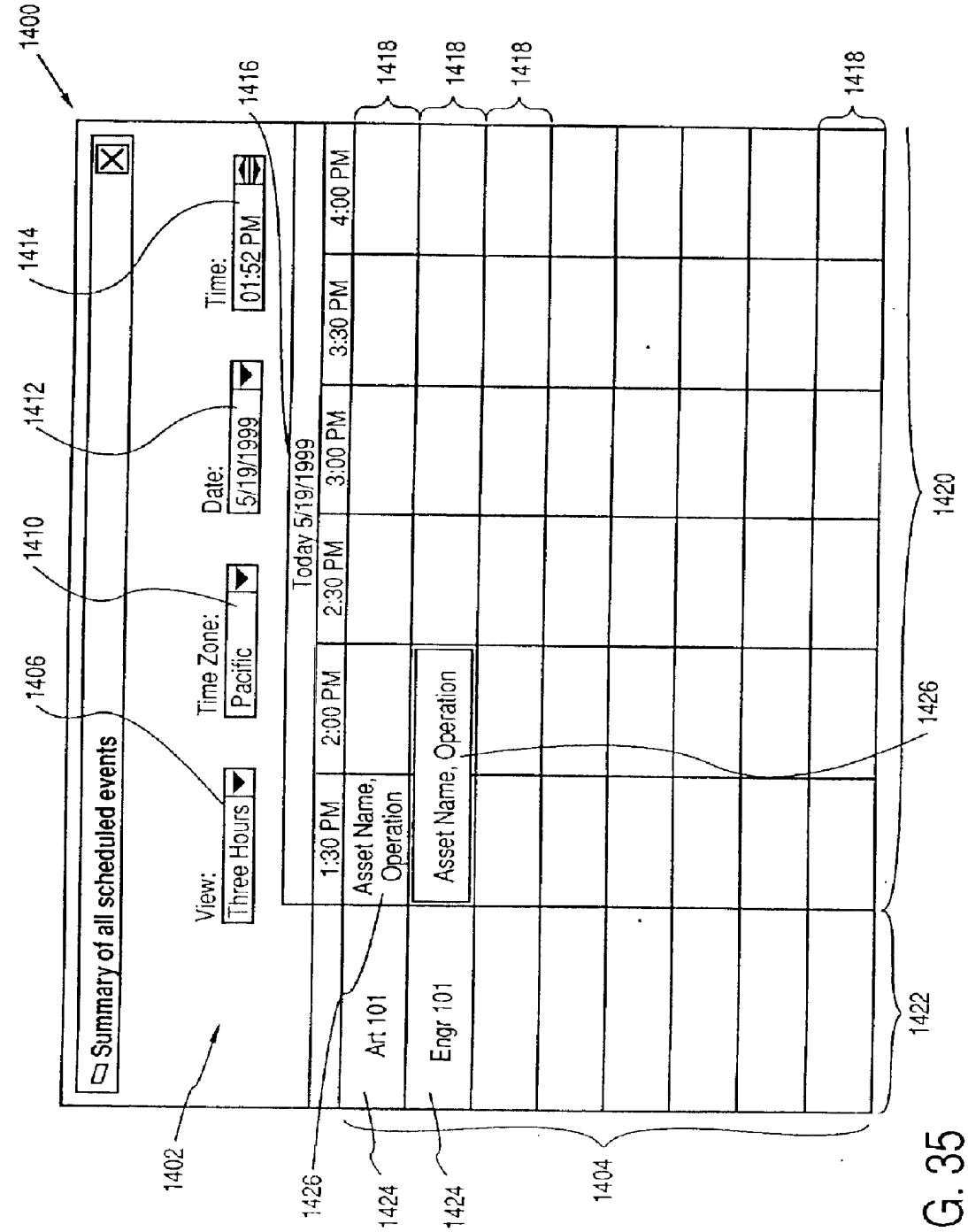

1400
Summary of all scheduled events
View:
Three Hours
Time Zone:
Pacific
Date:
5/19/1999
Time:
01:52 PM
Today 5/19/1999
1:30 PM
2:00 PM
2:30 PM
3:00 PM
3:30 PM
4:00 PM
Asset Name, Operation
Art 101
Engr 101
1402
1404
1406
1410
1412
1414
1416
1418
1420
1422
1424
1426

FIG. 35

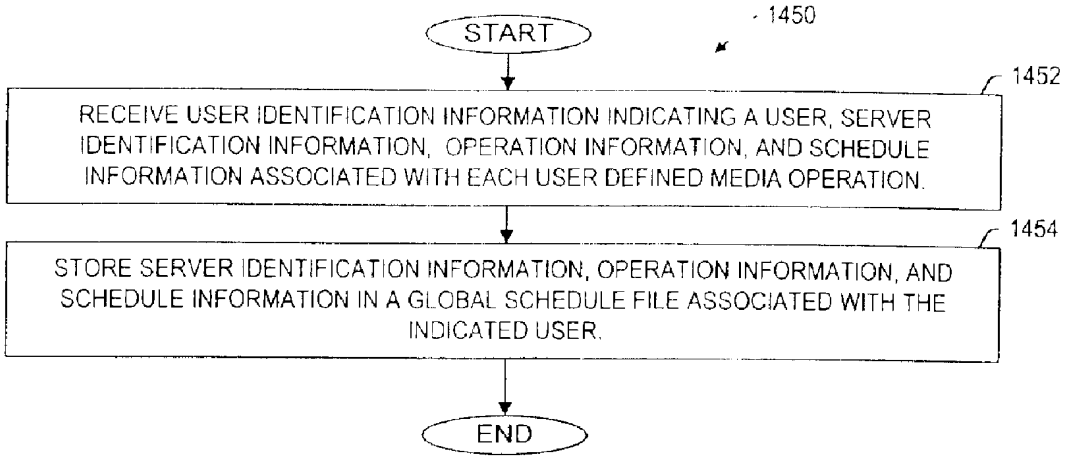

1450
START
1452
RECEIVE USER IDENTIFICATION INFORMATION INDICATING A USER, SERVER IDENTIFICATION INFORMATION, OPERATION INFORMATION, AND SCHEDULE INFORMATION ASSOCIATED WITH EACH USER DEFINED MEDIA OPERATION.
1454
STORE SERVER IDENTIFICATION INFORMATION, OPERATION INFORMATION, AND SCHEDULE INFORMATION IN A GLOBAL SCHEDULE FILE ASSOCIATED WITH THE INDICATED USER.
END

FIG. 36

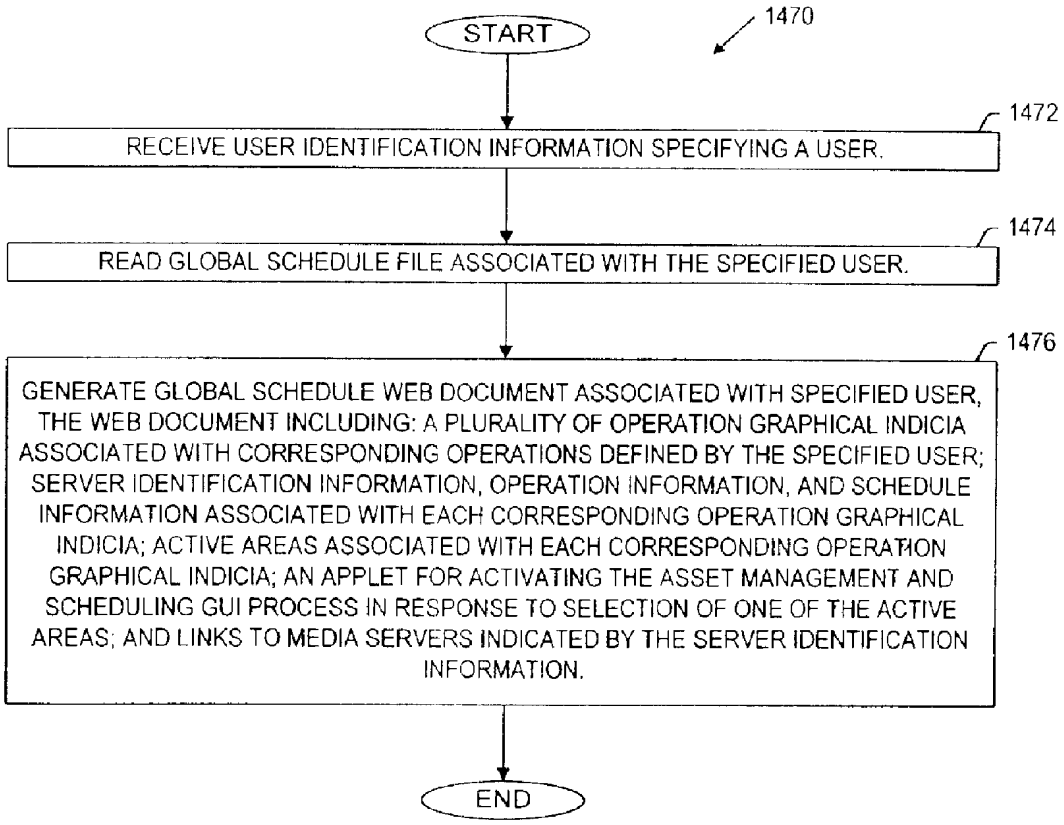

1470
START
1472
RECEIVE USER IDENTIFICATION INFORMATION SPECIFYING A USER.
1474
READ GLOBAL SCHEDULE FILE ASSOCIATED WITH THE SPECIFIED USER.
1476
GENERATE GLOBAL SCHEDULE WEB DOCUMENT ASSOCIATED WITH SPECIFIED USER, THE WEB DOCUMENT INCLUDING: A PLURALITY OF OPERATION GRAPHICAL INDICIA ASSOCIATED WITH CORRESPONDING OPERATIONS DEFINED BY THE SPECIFIED USER; SERVER IDENTIFICATION INFORMATION, OPERATION INFORMATION, AND SCHEDULE INFORMATION ASSOCIATED WITH EACH CORRESPONDING OPERATION GRAPHICAL INDICIA; ACTIVE AREAS ASSOCIATED WITH EACH CORRESPONDING OPERATION GRAPHICAL INDICIA; AN APPLET FOR ACTIVATING THE ASSET MANAGEMENT AND SCHEDULING GUI PROCESS IN RESPONSE TO SELECTION OF ONE OF THE ACTIVE AREAS; AND LINKS TO MEDIA SERVERS INDICATED BY THE SERVER IDENTIFICATION INFORMATION.
END

FIG. 37

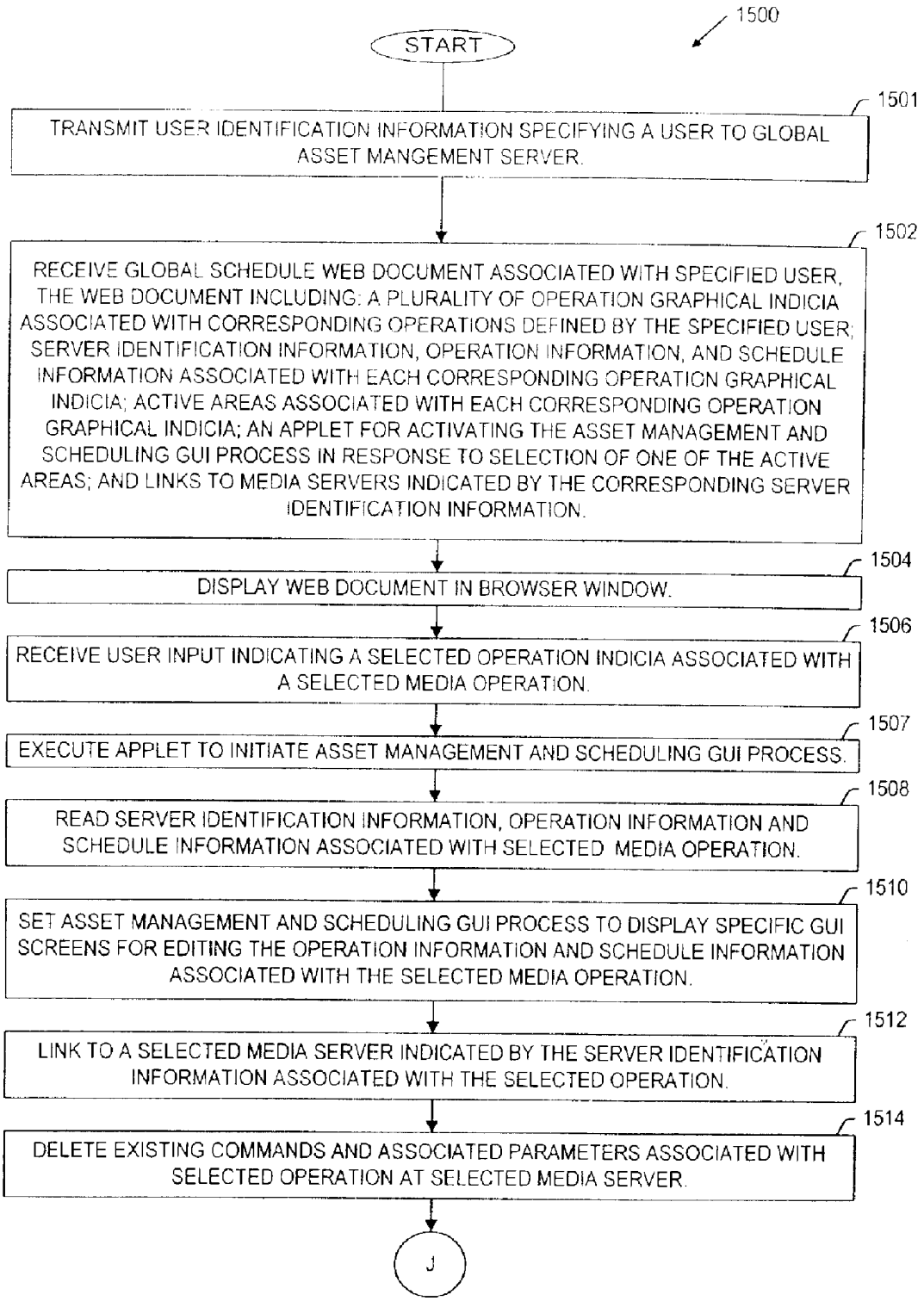

1500
START
1501
TRANSMIT USER IDENTIFICATION INFORMATION SPECIFYING A USER TO GLOBAL ASSET MANGEMENT SERVER.
1502
RECEIVE GLOBAL SCHEDULE WEB DOCUMENT ASSOCIATED WITH SPECIFIED USER, THE WEB DOCUMENT INCLUDING: A PLURALITY OF OPERATION GRAPHICAL INDICIA ASSOCIATED WITH CORRESPONDING OPERATIONS DEFINED BY THE SPECIFIED USER; SERVER IDENTIFICATION INFORMATION, OPERATION INFORMATION, AND SCHEDULE INFORMATION ASSOCIATED WITH EACH CORRESPONDING OPERATION GRAPHICAL INDICIA; ACTIVE AREAS ASSOCIATED WITH EACH CORRESPONDING OPERATION GRAPHICAL INDICIA; AN APPLET FOR ACTIVATING THE ASSET MANAGEMENT AND SCHEDULING GUI PROCESS IN RESPONSE TO SELECTION OF ONE OF THE ACTIVE AREAS; AND LINKS TO MEDIA SERVERS INDICATED BY THE CORRESPONDING SERVER IDENTIFICATION INFORMATION.
1504
DISPLAY WEB DOCUMENT IN BROWSER WINDOW.
1506
RECEIVE USER INPUT INDICATING A SELECTED OPERATION INDICIA ASSOCIATED WITH A SELECTED MEDIA OPERATION.
1507
EXECUTE APPLET TO INITIATE ASSET MANAGEMENT AND SCHEDULING GUI PROCESS.
1508
READ SERVER IDENTIFICATION INFORMATION, OPERATION INFORMATION AND SCHEDULE INFORMATION ASSOCIATED WITH SELECTED MEDIA OPERATION.
1510
SET ASSET MANAGEMENT AND SCHEDULING GUI PROCESS TO DISPLAY SPECIFIC GUI SCREENS FOR EDITING THE OPERATION INFORMATION AND SCHEDULE INFORMATION ASSOCIATED WITH THE SELECTED MEDIA OPERATION.
1512
LINK TO A SELECTED MEDIA SERVER INDICATED BY THE SERVER IDENTIFICATION INFORMATION ASSOCIATED WITH THE SELECTED OPERATION.
1514
DELETE EXISTING COMMANDS AND ASSOCIATED PARAMETERS ASSOCIATED WITH SELECTED OPERATION AT SELECTED MEDIA SERVER.
J

FIG. 38

(12)UK Patent (19)GB (11)2 353 877 (13)B

(45) Date of publication: **28.04.2004**

(54) Title of the invention: Service-oriented resource management across heterogeneous servers

(51) Int Cl[7]: **G06F** 9/50

(21) Application No: **0012035.2**

(22) Date of Filing: **19.05.2000**

(30) Priority Data:
(31) **09335274** (32) **17.06.1999** (33) **US**

(43) Date A Publication: **07.03.2001**

(52) UK CL (Edition W):
G4A AFN

(56) Documents Cited:
EP 0848334 A1
EP 0834809 A2
EP 0674280 A2
WO 1999/044121 A2
WO 1998/015903 A2
WO 1996/007257 A2
WO 1993/020511 A1
WO 1992/014205 A1

(58) Field of Search:
As for published application 2353877 A *viz:*
UK CL (Edition R) **G4A** AFN
INT CL[7] **G06F**
Other:
updated as appropriate

(72) Inventor(s):
Nelson R Manohar
Leon L Lumelsky

(73) Proprietor(s):
International Business Machines Corporation
(Incorporated in USA - New York)
Armonk, New York 10504,
United States of America

(74) Agent and/or Address for Service:
C J Ling
IBM United Kingdom Limited,
Mail Point 110, Intellectual Property Law,
Hursley Park, WINCHESTER, Hampshire,
SO21 2JN, United Kingdom

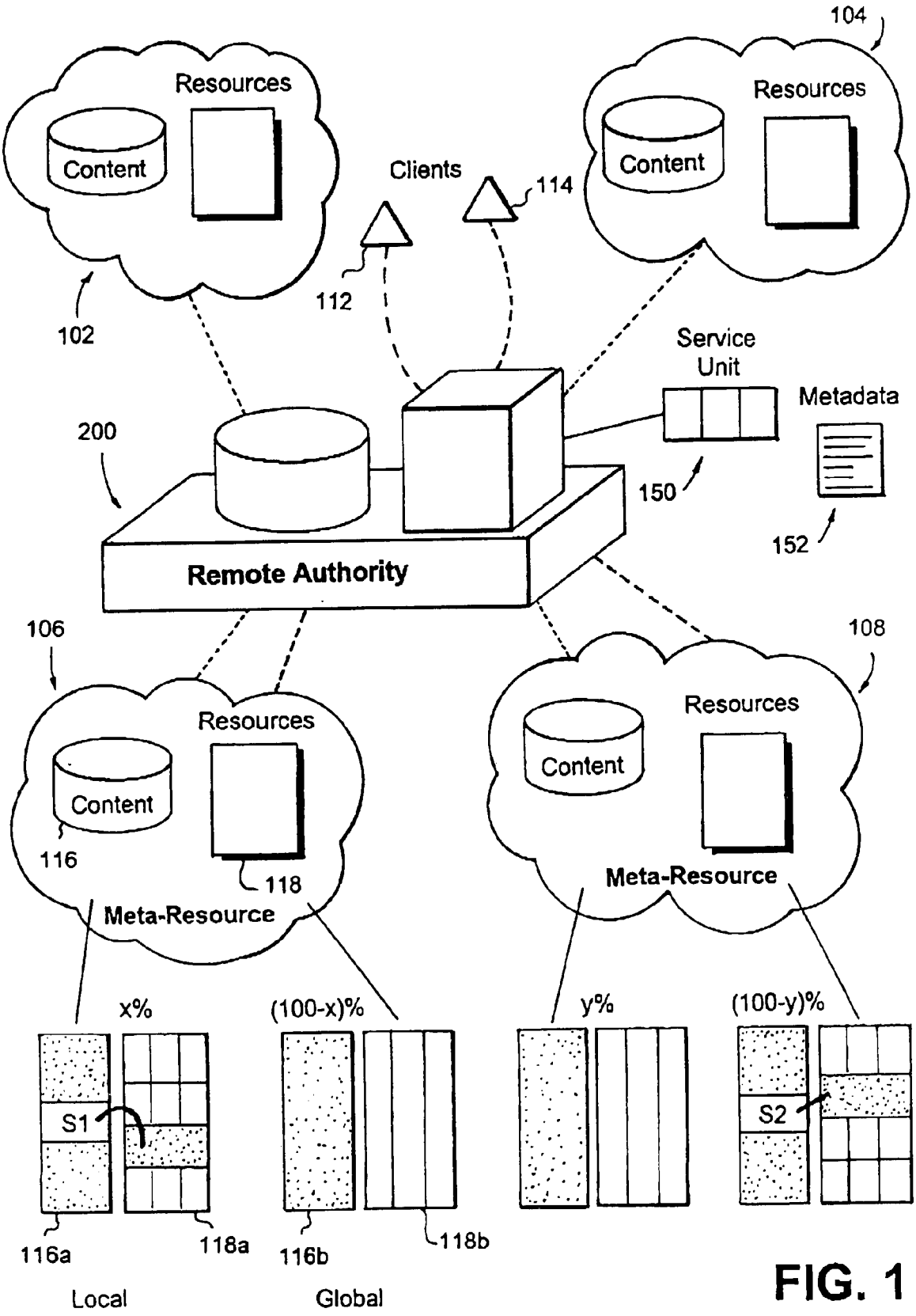
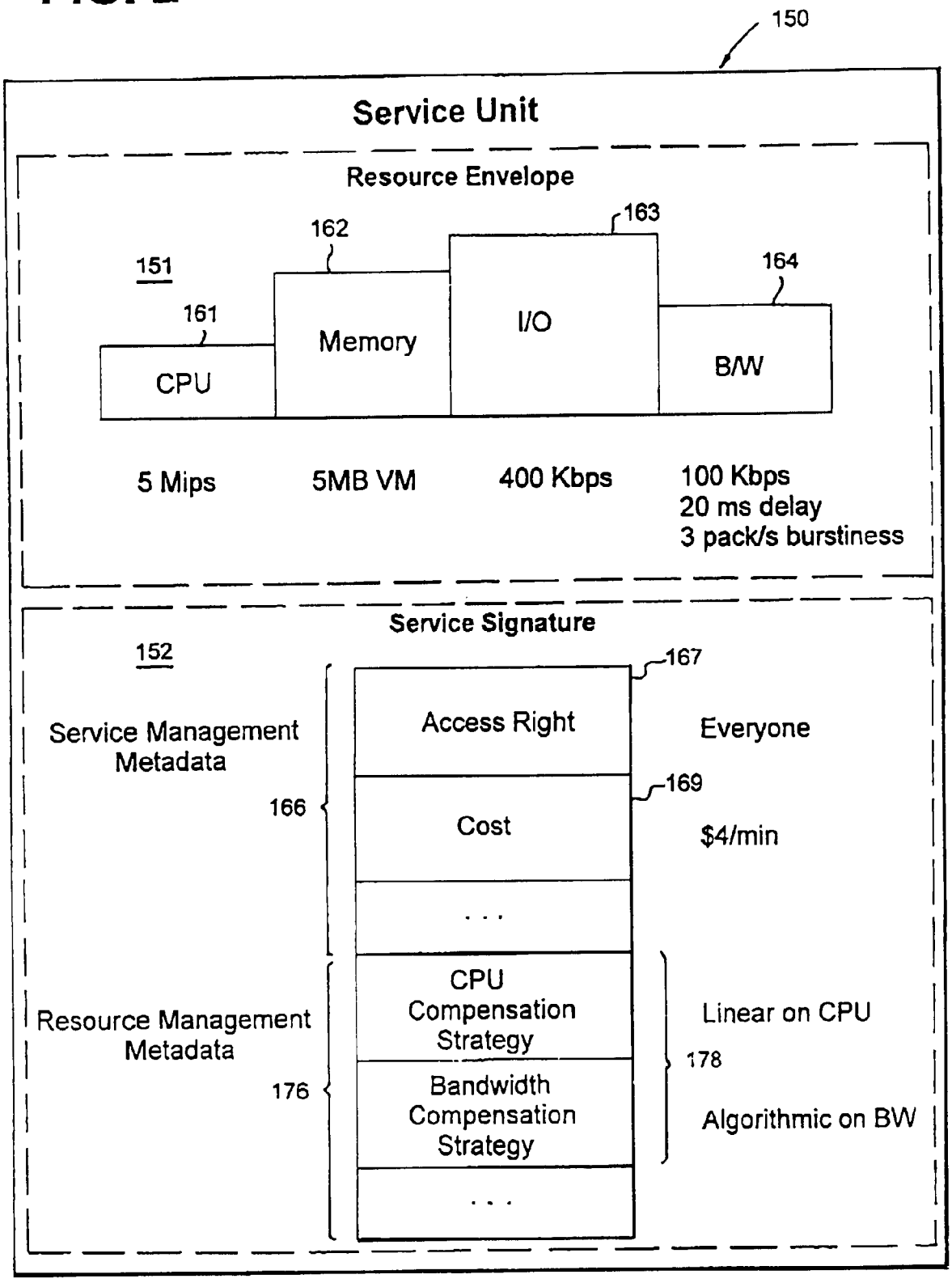

104
Resources
Content
Resources
Content
Clients
114
112
102
Service
Unit
Metadata
200
150
152
Remote Authority
106
108
Resources
Content
Resources
Content
116
118
Meta-Resource
Meta-Resource
x%
(100-x)%
y%
(100-y)%
S1
S2
116a
118a
116b
118b
Local
Global

FIG. 1

FIG. 2


150
Service Unit
Resource Envelope
151
161
CPU
162
Memory
163
I/O
164
B/W
5 Mips
5MB VM
400 Kbps
100 Kbps
20 ms delay
3 pack/s burstiness
Service Signature
152
Service Management Metadata
166
Resource Management Metadata
176
167
Access Right
Everyone
169
Cost
$4/min
. . .
CPU Compensation Strategy
Linear on CPU
178
Bandwidth Compensation Strategy
Algorithmic on BW
. . .

151

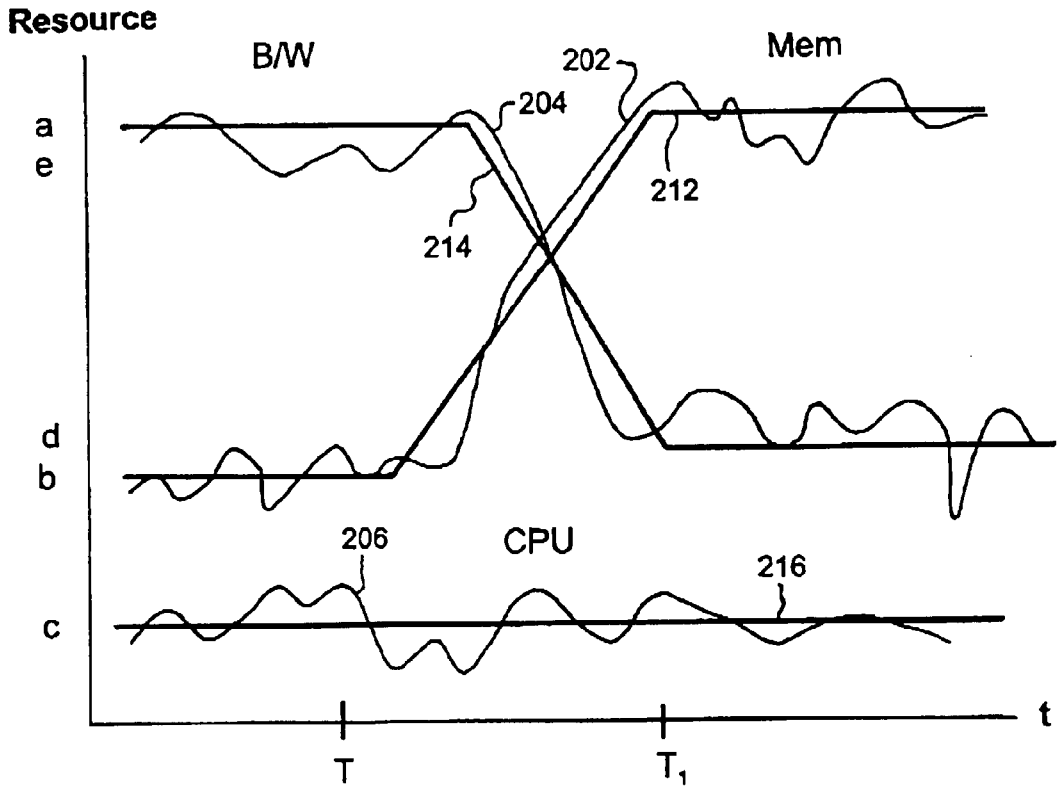


FIG. 3(a)

152

Resource Envelope

Time	CPU	Mem	B/W
T	c	b	a
T_1	c	e	d

FIG. 3(b)

FIG. 4

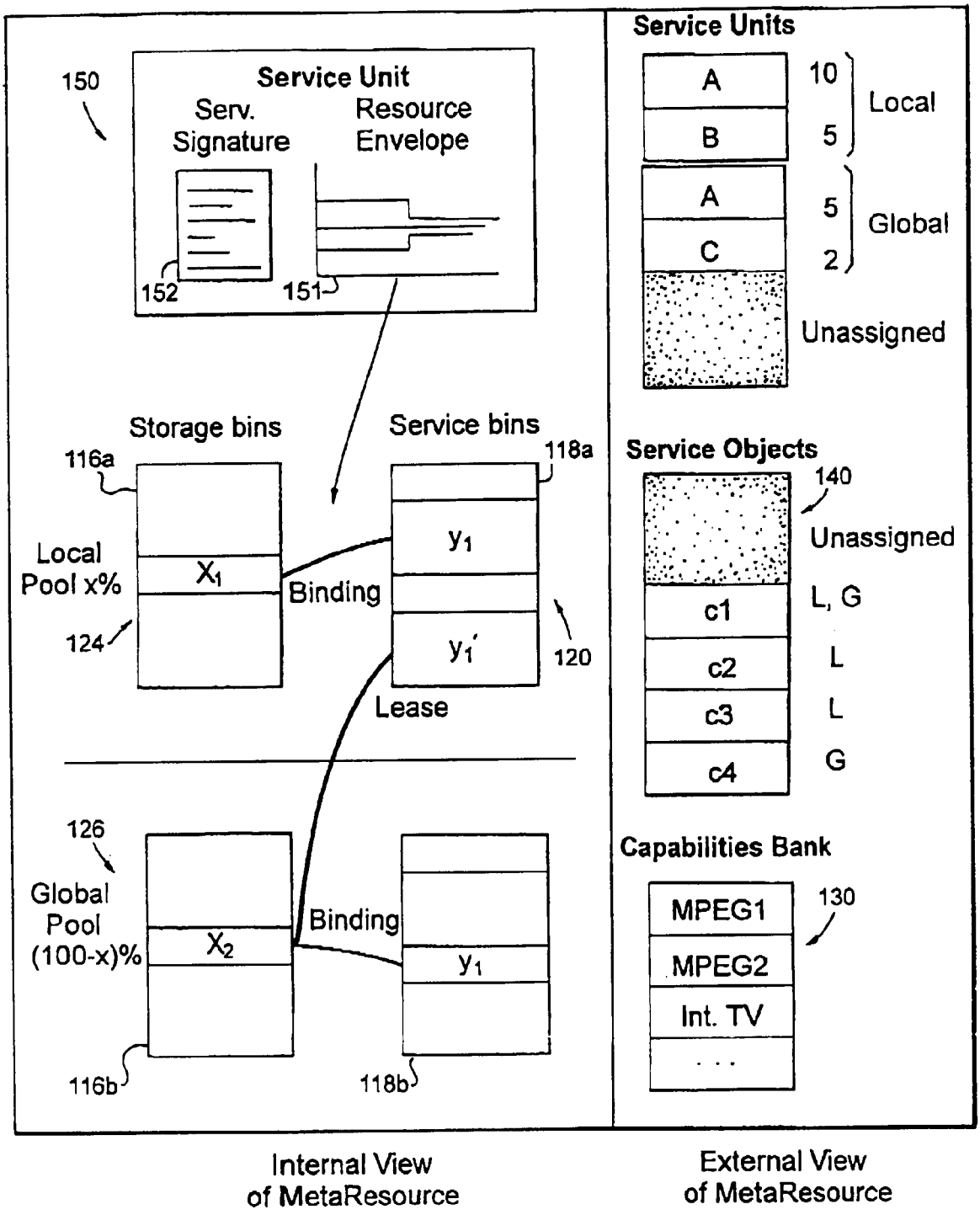
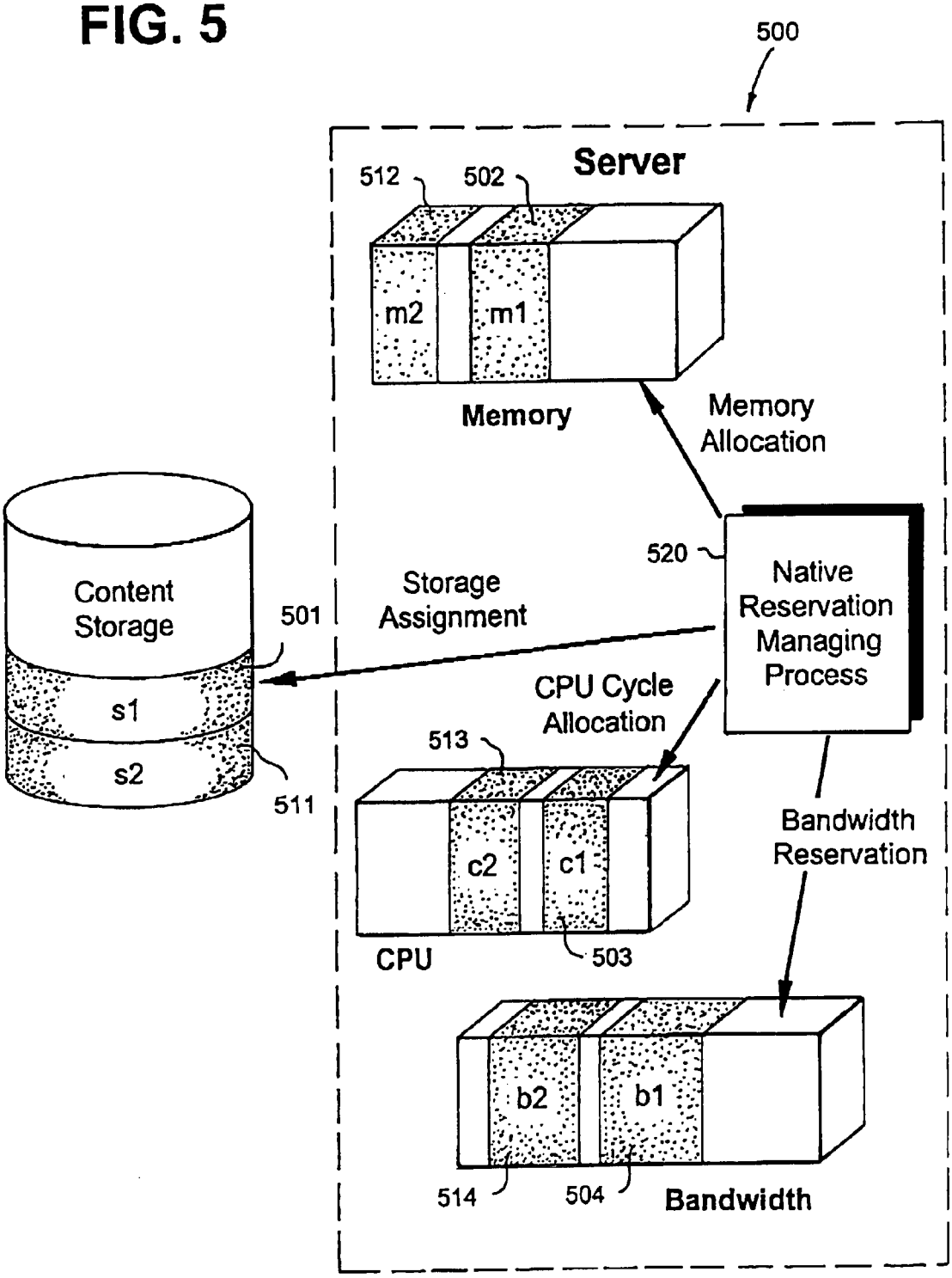
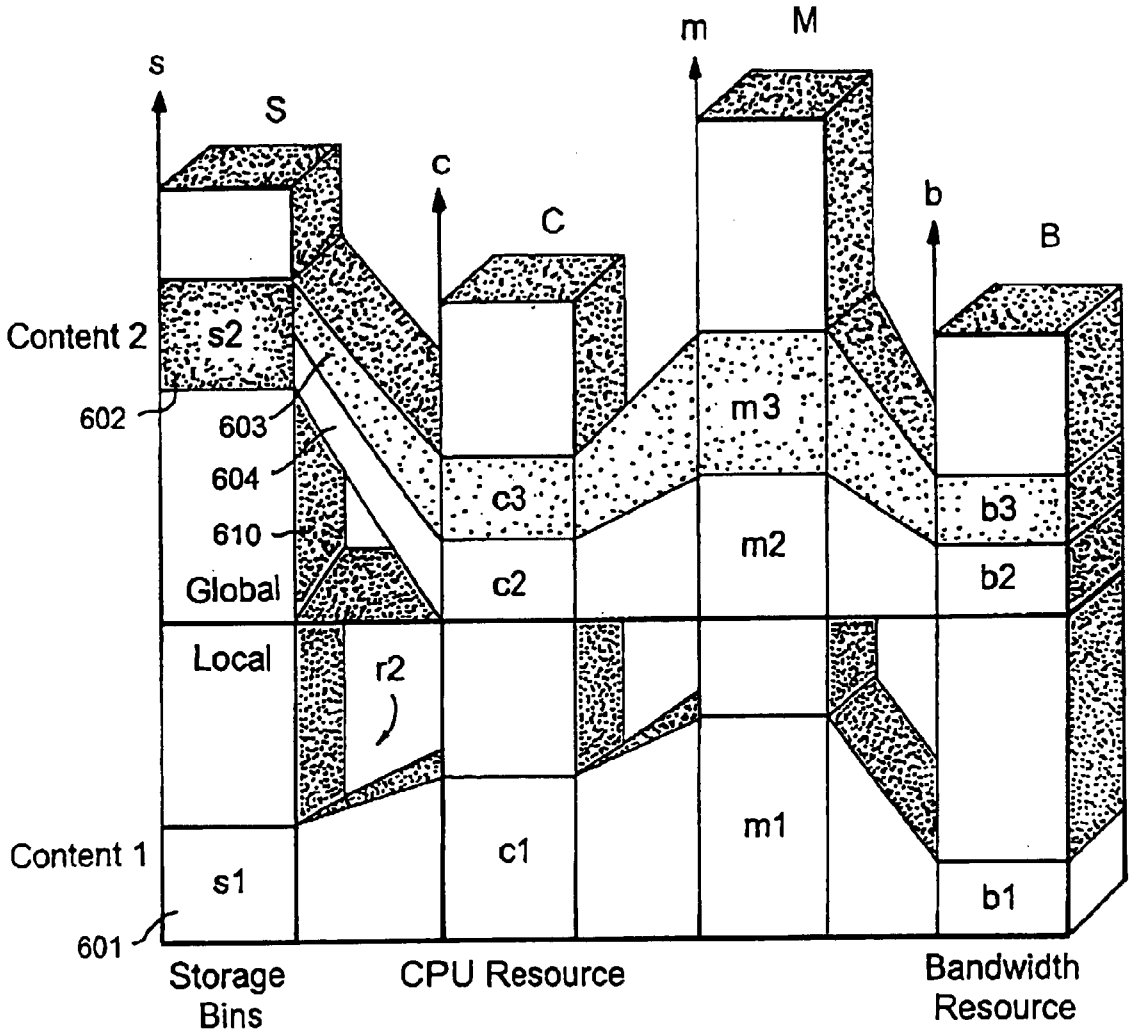

Service Units
150
Service Unit
Serv. Signature
Resource Envelope
152
151
A
B
10
5
Local
A
C
5
2
Global
Unassigned
Storage bins
Service bins
116a
118a
Local Pool x%
X_1
Binding
y_1
y_1'
124
120
Lease
Service Objects
140
Unassigned
c1
L, G
c2
L
c3
L
c4
G
126
Global Pool (100-x)%
X_2
Binding
y_1
Capabilities Bank
MPEG1
MPEG2
Int. TV
130
116b
118b
Internal View of MetaResource
External View of MetaResource

FIG. 5


500
Server
512
502
m2
m1
Memory
Memory Allocation
520
Native Reservation Managing Process
Content Storage
501
Storage Assignment
s1
s2
511
CPU Cycle Allocation
513
c2
c1
CPU
503
Bandwidth Reservation
b2
b1
514
504
Bandwidth



Note:
s1 is local storage bin,
s2 is global storage bin,
r1 (b1, c1, m1) is local service bin,
r2 (b2, c2, m2), and r3 (b3, c3, m3) are global service bins

FIG. 6

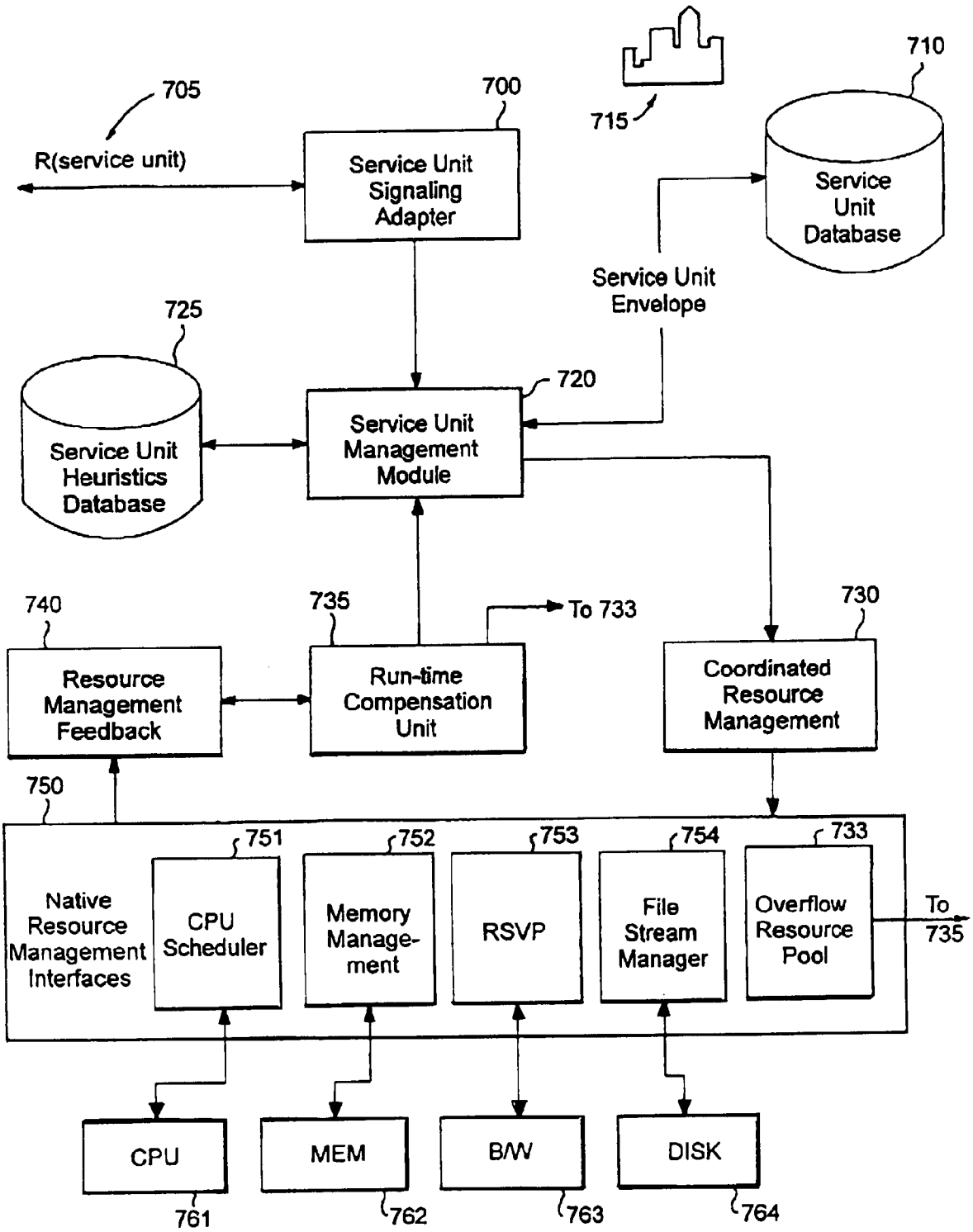

705
R(service unit)
700
Service Unit Signaling Adapter
715
710
Service Unit Database
Service Unit Envelope
725
Service Unit Heuristics Database
720
Service Unit Management Module
740
Resource Management Feedback
735
Run-time Compensation Unit
To 733
730
Coordinated Resource Management
750
Native Resource Management Interfaces
751
CPU Scheduler
752
Memory Manage-ment
753
RSVP
754
File Stream Manager
733
Overflow Resource Pool
To 735
CPU
MEM
B/W
DISK
761
762
763
764

FIG. 7

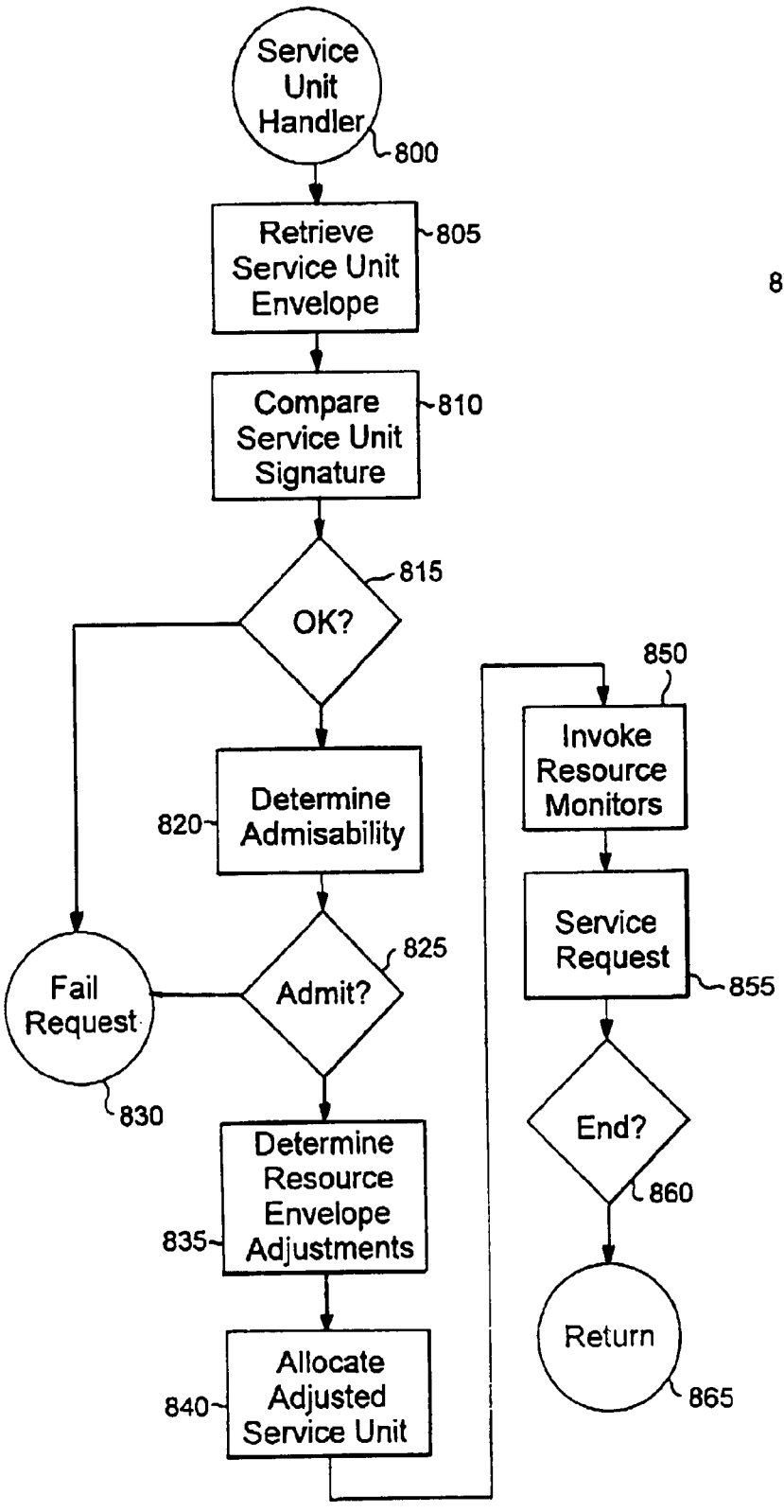

Service Unit Handler
800
Retrieve Service Unit Envelope
805
Compare Service Unit Signature
810
OK?
815
Determine Admisability
820
Admit?
825
Fail Request
830
Determine Resource Envelope Adjustments
835
Allocate Adjusted Service Unit
840
Invoke Resource Monitors
850
Service Request
855
End?
860
Return
865

FIG. 8(a)

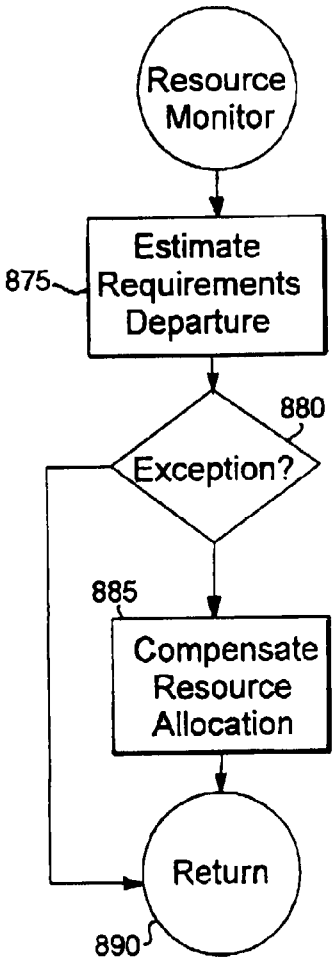

Resource Monitor
Estimate Requirements Departure
875
Exception?
880
Compensate Resource Allocation
885
Return
890

FIG. 8(b)

SERVICE-ORIENTED RESOURCE MANAGEMENT ACROSS HETEROGENEOUS SERVERS

Field of the Invention

The present invention relates generally to techniques for managing resources on distributed computer systems, and more particularly, to a system and method for managing resources for provisioning media services in future Internet utilities environments for the delivery of multimedia content.

Background of the Invention

The next generation of the Internet provides future applications with the opportunity to leverage resources across widely distributed resources via its ability to provide quality of service (QoS), large bandwidth connectivity (e.g., very-high-performance Backbone Network Service or vBNS), differentiated services (diff-serv), and unique addressing (e.g., IPv6). To applications, this blend of technologies introduces the notion of virtual proximity, i.e., the logical and apparent proximity of widely distributed resources to applications. For applications to make efficient use of these resources, mechanisms are needed to distribute the resulting uncoordinated application load presented by millions such applications to these resources. However, the task of resource management across widely distributed media servers across the Internet is not yet well understood and has been recognized as an emergent issue in the literature.

At the same time, an emerging set of computer utilities is foreseen. Some examples of these new service utilities are concepts such as Internet TV and radio, to name a few. The nature of a service utility is centered on the engineering of distribution channels. In traditional distribution channels such as the telephone network, control of the distribution is achieved via over-engineering and statistical multiplexing of resources. Until now, several reasons have discouraged over-engineering of resources over the Internet -those being primarily, medium properties at high load (e.g., burstiness and unpredictability) and, the lack of a central administration to enable the over-engineering of end-to-end resources. With the emergence of virtual proximity, a new paradigm is possible in which statistical over-engineering of end-to-end resources is now desirable.

The task of leveraging the increased availability of widely distributed content and resources becomes very important with the proliferation of the next generation of the Internet, e.g., Internet2. There are a number of publications and patents in the area of QoS-driven resource management. Most of the work has been focused on either the network, as described in U.S. Patent 5,388,097 issued February 7, 1995 to Baugher, M.J. et al.,and entitled "System and Method for Bandwidth Reservation for Multimedia Traffic in Communication Networks," and U.S. Patent 5,581,703 issued December 3, 1996 to Baugher, M.J. et al, and entitled "Method and Apparatus for Reserving System Resources to assure Quality of Service"; or, the operating system, such as described in the reference "An Architecture Towards Efficient OS Support for Distributed Multimedia", Proceedings of IS&T/SPIE Multimedia Computing and Networking Conference '96, San Jose, CA, January 1996 by David K.Y. Yau and Simon S. Lam. With the proliferation of multimedia services on Internet, it was soon realized that while IP networks were able to provide a simple, best-effort delivery service, the IP protocol is not suited for use with new real-time applications, such as multimedia streaming, Virtual Reality applications, distributed supercomputing. As a result, new network protocols, such as Resource Reservation Setup Protocol (RSVP) (See, e.g., "The Grid: Blueprint for a New Computing Infrastructure," Edited by Ian Foster and Carl Kesselman, Chapter 19, pp. 379-503, Morgan Kauffman Publishers, 1999); Real Time Transport Protocol (RTP); Real Time Transport Control Protocol (RTCP) and others, were developed (See, e.g., William Stallings, "High-Speed Networks: TCP/IP and ATM Design Principles", Prentice Hall, 1997; and, I. Busse, B. Deffner, and H. Schulzrinne, "Dynamic QoS Control of Multimedia Applications based on RTP", Computer Communications, January 1996), enabling applications to request and negotiate network QoS parameters, such as bandwidth and latency. Deployment of those protocols on the current Internet has not been successful, firstly because it required upgrading all the non-RSVP routers and servers system software. Secondly, even if RSVP were deployed on the current Internet, very limited bandwidth and computing resources would still have been the bottleneck for successful deployment of real-time applications. The current Internet was built on the backbone, enabling cross-country communications on relatively unclogged T3 (45 megabit per second). Proliferation of graphic pages, and streaming audio and video applications depleted those resources quite fast. Even worse, the rate of user's population growth is considerably higher than newly build network resources.

The National Science Foundation and MCI Corporation, responding to the emerging needs of Internet community has been building a new network, called the very-high-performance Backbone Network Service (vBNS). This nationwide network also provides a backbone for the two foundations, university-led effort called Internet 2 and by federal research agencies, called New Generation Internet. The vBNS allows most of the connected institutions to run at 622 million bits per second (OC12). By the year 2000, vBNS is expected to operate at 2.4 gigabits per second (2,400 megabits per second) by the year 2000.

The vBNS system exploits RSVP protocol to support two distinct classes of services: a Reserved Bandwidth Service, i.e. a service with bandwidth commitment, and a traditional best-effort IP service (See, e.g., Chuck Song, Laura Cunningham and Rick Wilder, "Quality of Service Development in the vBNS", MCI Communications Corporation, provided at the URL http://www.vbns.net/presentations/papers/QoSDev/ieeeqos.htm. Still, resource management at the network layer for vBNS is done separately from operating system layer and in isolation from application needs and availability of the end-resources, such as storage and computing resources.

A new breed of high performance applications such as remote surgery, robotics, tele-instrumentation, automated crisis response, digital libraries of satellite data, distance learning via multimedia supported Web sites, enhanced audio, and video, is emerging. However, to accommodate such high performance applications and their continuous media flows, it is not enough to increase or reserve network capacity. These new applications require end-to-end resource reservation and admission control, followed by co-ordination of distributed functions such as: (a) resource scheduling (e.g., CPU, disk, etc.) at the end-system(s), (b) packet scheduling and flow control in the network, and (c) monitoring of the delivered end-to-end quality of service. It is essential that quality of service is configurable, predictable and maintainable system-wide, including the end-system devices, communications subsystem, and networks. Furthermore, all end-to-end elements of distributed systems architecture must work in unison to achieve the desired application level behavior.

Up to date, there has been considerable effort in the development of end-to-end quality of service support. Among them are Heidelberg QoS Model, developed within HeiProject at IBM's European Networking Center and described in the reference entitled "HeiRAT - Quality of Service Management for Distributed Multimedia Systems", Multimedia Systems Journal, 1996 by

Volg, C., Wolf, L., Herrtwich, R. And H. Wittig; an Extended Integrated Reference Model (XRM), developed by COMET group at Columbia University such as described in the reference entitled "Building Open Programmable Multimedia Networks", Computer Communications Journal, Vol. 21, No. 8, pp. 758-770, June 1998 by Campbell, A.T., Lazar, A.A., Schulzinne, H. And R. Stadler; OMEGA end-point architecture, developed as the interdisciplinary research effort in the University of Pennsylvania such as described in the reference entitled "Design, Implementation and Experiences of the OMEGA End-Point Architecture", Technical Report (MS-CIS-95-22), University of Pennsylvania, May 1995 by Nahrstedt K. And J. Smith; in-serv Architecture which is a contribution of the Internet Engineering Task Force (IETF) such as described in the reference entitled "A Framework for End-to-End QoS Combining RSVP/Intserv and Differentiated Services," Internet Draft, IETF, March 1998 by Bernet Y, et al.; the Quality of Service Architecture QoS-A, developed by A. Campbell, and presenting an integrated framework dealing with end-to-end QoS requirements such as described in the reference entitled "A Quality of Service Architecture", PhD thesis, Lancaster University, January 1996 by Andrew T Campbell. Another reference which analyzes the above mentioned QoS paper is entitled "A Survey of QoS Architectures", ACM/Springer Verlag, Multimedia Systems Journal, Special Issue on QoS Architecture, Vol. 6, No. 3, pp. 138-151, May 1998 by Aurrecoechea, C., Campbell, A.T. and L. Hauw.

Substantial work has been done by SRI International, developing an End-to-End Resource Management of Distributed Systems (ERDoS), which enables adaptive, end-to-end, scalable resource management of distributed systems such as described in the reference ERDOS QoS Architecture, Technical Report, SRI International, May 1998. An extensible Resource Specification Language (RSL) and the resource management architecture has been implemented within Globus meta-computing toolkit, and used to implement a variety of different resource management strategies such as described in Czajkowski, K., et al., "A Resource Management Architecture for Metacomputing Systems" Proc. IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing, 1998; and Foster, I., Kesselman, C., "The Globus Product: A Status Report" Proc. IPPS/SPDP '98 Heterogeneous Computing Workshop, pp. 4-18, 1998.

While the architectures described in the above-mentioned references are directed resource reservation and management of end-to-end resources, they generally assume a single, even geographically limited network subsystem which provides bounds on delay, errors and meet bandwidth

demands, and an operating system which is capable of providing run time QoS guarantees. However, the next generation Internet must be viewed not as only a network of networks, but first and foremost a system of distributed systems. In this paradigm, not only the communication resources, but also the computing and storage servers are shared among many users.

Thus, the architectures mentioned above do not provide a coordinated management of overall system resources as a function of request activities for individual content and computing resources. It deals with resources pre-assigned to particular services. Consequently, quality of service must be degraded in response to growing volume of requests for such services over and above an established limit. As the above-mentioned architectures focus on providing QoS as requested by application, they do not take an advantage of a possible aggregation of resources due to commonality between user requests for a particular service.

For example, it would be desirable to determine commonality for the usage history of a particular multimedia content, e.g., bursts of requests within short time intervals, the proximity of origination addresses of requests, etc. In addition, the architectures described above do not allow for dynamic monitoring and recording of resource consumption for individual services as well as for groups of related services, with the purpose of calculating cost of service for individual clients.

Thus, it would be highly desirable to provide a resource management mechanism for widely distributed resources that leverages the strengths of the next generation of Internet, and further, a system that enables coordination of all system resources for dynamic polling, re-assignment and release of resources, consequently, to provide a QoS requested by an application.

Prior approaches to resource management had focused on the independent optimization of individual resources in an operating system -whether centralized or distributed. Later approaches have provided higher level models for independent resource management but lack the coordination of multiple resources. For example, although RSVP provides end-to-end resource reservation as required by multimedia applications, it lacks the notion of coordination and co-allocation across resources such as CPU or disk.

Thus, it would be highly desirable to provide a novel resource management system that moves away from single resource, single node resource management to multiple resource, multiple node resource management.

Disclosure of the Invention

Accordingly, the present invention provides a system and method for managing resources in a distributed network that supports a utilities model for media services and accounts for virtual proximity over the next generation Internet. In an Internet multimedia utilities model, media services are provisioned to a paying subscribers mass in a manner such that the location of the provisioning party is irrelevant to subscribing parties as long as guarantees are made and met.

It is a benefit of the present invention to provide a mechanism for managing resources across heterogeneous servers that is capable of driving and fostering the Internet media utilities model, and particularly, a mechanism for administering, deploying, and configuring media servers in an Internet media utilities environment that permits leveraging virtual proximity over the resources provided by distributed media servers scattered throughout a network.

It is another benefit of the present invention to implement a novel approach to resource management by moving away from single resource, single node resource management to a multiple resource, multiple node resource management. This novel approach to resource management moves away from the optimization of single resource utilization to the statistical engineering of a set of resources. The present invention implements means to integrate resource management (i.e., typically low-level functionality) with high-level service management (i.e., application-level functionality).

Preferably, the present invention relates to a service-oriented approach to multimedia resource management across distributed servers for the provisioning of Internet multimedia utilities to paying subscribers. Particularly, a system and method for the integrated management of services and resources across widely distributed media servers is provided. The system and method includes configuring heterogeneous media servers in terms of homogeneous service-oriented resource units referred to as "service units" which are used to represent a resource allocation commitment from a participating server to be able to provision a particular media service on-demand. In the preferred embodiment of the present invention, the service unit is represented in terms of an allocation vector whose members represent resource allocations for memory, disk, network, and CPU

resources. A service unit represents an envelope of resource requirements as needed for provisioning of a media service with the generated resource envelope representing a bound set according to some criteria over a set of critical resources required for the provisioning of a media service. Accordingly, a different service unit is associated with each service with the same service unit definition possibly resulting in different resource envelopes across different servers supporting the same media service. Such resource envelope profiles may be time-variant or constant. The system of the present invention additionally provides a mechanism for generating and adapting such resource envelopes for any media service.

According to the present invention, a server is referred to as a meta-resource ("a resource that provides resources") which refers to the service viewpoint and not the traditional reference to low-level resources such as memory, CPU, bandwidth, etc.. The ability (e.g., the necessary software) of any meta-resource to provision a media service is referred to as a capability with the notion of the service unit being the commitment to provision such service to a subscribing party.

It is also an advantage of the present invention to allow the meta-resource to remain autonomous. Thus, according to the principles of the present invention, by providing application-level access control onto a meta-resource, the autonomy of meta-resources is preserved. To this end, each service unit is associated with metadata referred to as a "service signature" which is implemented to customize the service commitment of a meta-resource, e.g., by delivering hints to the meta-resource about resource management. For example, the service unit could be used to define access rights and characteristics for any particular service unit. Similarly, the service signature may recommend run-time compensation strategies to be used to update the resource envelope for this service unit under this meta-resource type at different loads. Thus, the service signature is one of the ways in which the present invention allows the integration of service management with resource management.

Advantageously, the present invention enables an arbitrary media server to participate and offer a position of value to an open and deregulated Internet utilities environment. To this end, the system of the

present invention removes heterogeneity in resource management across heterogeneous servers. That is, according to the present invention, the meta-resource may react to a remote administration authority or entity however, retain autonomous control of its resources. Thus, the present invention permits the configuration of a meta-resource in terms of service units (i.e., service-oriented promises or commitments) to the authority while the fulfillment of these commitments is left to the meta-resource.

Furthermore, according to the present invention, a meta-resource may be partitioned and administered as different and independent resource pools: a local resource pool implemented by the meta-resource to provide media services under local administration; and, a global resource pool implemented by the meta-resource to provide media services under remote administration. Administration of these resource pools is enabled in terms of the type and number of service units, thus enabling the administrator of the meta-resource to enhance the value such meta-resource offers to the remote administration. Such administration may be based on some criteria such as the expected revenue, or some other cost metric. This results in individual meta-resources being entitled to various degrees of greediness toward acquisition of service requests by administration of their resource pools and their selection of service units and service capabilities. The present invention is used to address the domain of Internet utilities and its application to multimedia services in a novel and useful manner. As an emerging set of Internet utilities is foreseen including Internet TV and Internet radio, to name a few, with applications ranging from distance learning, virtual reality, entertainment services, etc., the nature of a service utility is centered on the engineering of distribution channels. In traditional distribution channels such as the telephone network, control of the distribution is achieved via over-engineering and statistical multiplexing of resources. With the emergence of virtual proximity, the present invention makes possible and desirable the statistical over-engineering of end-to-end resources while providing a new revenue participation model for meta-resources while fostering the proliferation of meta-resources.

Brief Description of the Drawings

The present invention will now be described, by way of example only, with references to preferred embodiments thereof, as illustrated in the following drawings:

Figure 1 is an illustration depicting an Internet media utilities environment.

Figure 2 illustrates the service units construct as comprising a service signature and a resource envelope.

Figures 3(a) and 3(b) illustrates the concept of generating a resource envelope portion of a service unit.

Figure 4 depicts the relationships between service units, service bins, storage bins, global and local pools, resource envelopes, and service signatures within a meta-resource according to the invention.

Figure 5 depicts the allocation of individual resources on a meta-resource and its relationship to service and storage bins according to the invention.

Figure 6 illustrates via an example the use of service and storage bins and the use of global and local bins in a meta-resource as prescribed in the present invention.

Figure 7 is an illustration depicting the component architecture of the service unit resource management module at a meta-resource as prescribed in the present invention.

Figures 8(a) and 8(b) are flow charts depicting the method for handling a service request by the service unit resource management module.

Detailed Description of the Invention

Figure 1 depicts an Internet media utilities environment model or global meta-system 100 where media services are provisioned to a paying subscribers mass. According to the present invention, as will be described hereinafter in greater detail, the location of the meta-resources, e.g., provisioning servers (102, 104, 106, 108) is irrelevant to subscribing parties e.g., clients (112) and (114), as long as guarantees are made and met by implementing suitable resource management techniques. The present invention is useful and suitable for the next generation of the Internet where due to virtual proximity, resources will be appear to be abundant by efficiently managing scattered resources from around a wide area network.

As resource management must deal with widely distributed and heterogeneous systems each with varying approaches to resource management, the techniques of the present invention leverages awareness of a common service model as a mechanism to address heterogeneity across servers. To this end, the present invention facilitates configuration and management of resources in a server in a platform-independent manner. It is essential and desirable that the remote administration of the global meta-system, as provided by an remote authority (200), to be described in greater detail herein, be able to compare heterogeneous servers, e.g., servers (102) and (104) on the same basis when choosing between then. To this end, the present invention implements a service unit construct (150) with the capacities of a meta-resource (i.e., a server) being defined in terms of service units.

Particularly, as illustrated in Figure 2, the service unit (150) is represented in terms of an allocation vector (151) whose members represent resource allocations for memory (162), disk I/O (163), network bandwidth B/W (164), and CPU resources (161). As will be described in greater detail herein, the service unit allocation vector is referred to herein as a "resource envelope" as needed for provisioning of a media service. The resource envelope represents a bound set according to some criteria over critical-resource requirements associated with the provisioning of a media service. Further, the remote authority maintains a "service signature" or meta-data construct (152) relating to the characteristics of an associated resource envelope. In general, a service signature comprises contains service management metadata (166) as well as resource management metadata (176). It is a preferred aspect of the present invention that such service signature be used to customize the integrated service-oriented management approach to resource management. In particular, a service signature describes information, hints, and recommendations about things such as access rights, privileges and characteristics of services that can use that particular service unit. For example, the service unit could be used to define access rights (167) and cost (169) characteristics for any particular service unit. Preferably, the service signature is used as well to deliver hints to the meta-resource about resource management. For example, the service signature could be used to recommend run-time compensation strategies (178) to be used to update the resource envelope for a service unit provided for a given meta-resource type at possibly different loads. That is, as will be described in greater detail herein, the system of the present invention provides means to compensate for differences between actual resource requirements found during the

provisioning of a media service and the resource profile associated with a service unit. It is a possibility that the resource envelope projected by a service unit may incorrectly estimate the resource requirements needed to provision the service object. The service signature (150) is one of the ways in which the present invention allows the integration of service management with resource management.

According to the present invention, the remote authority (200) implements a mechanism for configuring heterogeneous meta-resources in terms of homogeneous service-oriented resource units, i.e., the service units. Particularly, the service unit is used to represent a resource allocation commitment from a participating server to be able to provision a particular media service on-demand. A different service unit is associated with each service and that the same service unit definition may result in different resource envelopes across different meta-resources supporting the same media service. Such resource envelope profiles may time-variant or constant as described within, as will hereinafter be described in greater detail. It is a preferred aspect of the present invention to generate and adapt such resource envelopes for any media service.

As will be described in greater detail herein, the present invention implements the construct of service bins as a mechanism to enable the co-allocation of resources in a meta-resource. Generation of a service bin is intended to provide a performance envelope over the overall resource requirements for provisioning of a service. It should be noted that the generation of a resource envelope may be done either at the meta-resource or by a remote party. Either way, techniques are known in the art for generating a performance envelope for various resources. For example, administrators may generate a bin specification by applying a test suite to an application over typical content to generate a resource envelope. The envelope is intended to define a long-term resource allocation that may be, for example, a worst case or a 70th percentile. If a worst case allocation is not used, then there are techniques to, on-demand, compensate for the difference on resource allocation requirements. In particular, in the present invention, a shared overflow pool is used to allot such resources on-demand. A ranking of deviations may be used to further optimize the use of this overflow pool, as for example, to set aside such deviations ahead of time. A ranking of deviations may be produced to allow the compensation bin to set aside such deviations ahead of time.

In general, the generation of a resource envelope for a particular service is comprised of the following steps: first, an initial set of service objects is selected according to some criteria such as frequency of use or similarities; second, a resource profile is obtained for each critical resource associated with the provisioning of the service object; third, the resource profile is smoothed into K fixed duration intervals; fourth, for each interval, a resource requirements envelope is calculated. A skilled artisan will appreciate that there are known ways to generate such smoothed performance envelopes for stored continuous media. For example, for an internet TV model based on the streaming of stored MPEG content, there are known ways to retrieve the profile comprising of, at most, K segments; fifth, the extraction of meta-data relating to the characteristics of the envelope and its fit over the representative service objects. For example, an 80% percentile may be used to generate a 2nd-tier envelope where peaks about the 80% are identified on the metadata; finally, storing the resource envelope and meta-data into the service unit database.

Figures 3(a) and 3(b) particularly illustrate by way of example, the test generation of a time-variant resource envelope profile (151) (Figure 3(a)) which is essentially a plot of the resource usage versus time for a given service, e.g. provision of steaming video. As shown in Figure 3(a), one method determines variance 202 of a memory resource, which as shown in Figure 3(a), increases from usage level "b" (e.g., 2 MBytes) at time T to a usage level "e" (e.g., 6 MBytes) at time T2. Likewise, as depicted by line 204, a B/W resource for the same service decreases from usage level "a" at time T to usage level "d" at time T2. The CPU performance variance is shown as being constant at level "c" over the time interval T to T1. The respective dark lines 212, 214 and 216 represent the average or static value for that resource over time, or, may be defined to represent a worst-case level. A corresponding table (152) of the time-variant resource envelope shown in Figure 3(a) is depicted in Figure 3(b).

It is understood that different meta-resources may have slightly different resource envelopes for the same service unit to suit their particular needs and requirements. Furthermore, a service unit database (not shown) enables caching, storage, and updating of resource envelopes for any particular service unit.

Referring back to Figure 1, there is further illustrated the concept of a meta-resource, i.e., server, which is represented in terms of capabilities, which, for example, may represent a heterogeneous media unit

streaming capacity. Each server, e.g., server 106, may be logically partitioned as comprising a storage bin (116), for maintaining persistent content e.g., a multi-media object, and storage bin (118) for maintaining service resources, e.g., for streaming multimedia content to a client. Each server additionally may be logically partitioned as comprising local storage and service capabilities (116a, 118a) and global storage and service capabilities (116b, 118b). For example, the server (116) of Figure 1 illustrates local storage object S1 being bound with a local resource to provide, for example, digital video a streaming service. Each capability may be assigned to local or global management with a remote administrator (200) managing the global capabilities, for example, while local capabilities are managed by the administration of the meta-resource. It should be understood that the administration of the meta-resource allocates a ratio of global (e.g., 100 - x %) and local (e.g., x %) capabilities.

As shown in Figures 1 and 4, the capacity of a meta-resource is measured in terms of service units (150) (including resource envelopes (151) and a service signature (152) comprising the metadata), a local pool (124) comprising storage bins (116a) and storage bins (118a), and a global pool (126) comprising storage bins (116b) and storage bins (118b). For each service unit, a service bin may be associated which has one slot (e.g., 120) for each different resource capability tuple, e.g., memory, CPU, B/W, etc. In addition to capabilities, a meta-resource manages a storage bin which is used for each content. Each meta-resource is assigned content that is local and/or global. Replicated content is globally maintained. Local content is referred to as original content. A streaming connection is formed by binding a storage bin to a service bin, the binding being evaluated at a quality of service. The service bin is associated with resource requirements needed to stream content in such resolution. Each content is associated with a resource profile (151) which is used to facilitate the evaluation of the binding.

The resource management system of the present invention opens such Internet media utilities model to arbitrary media servers. In particular, the system of the present invention enables an arbitrary media server to participate and offer a position of value to an open and deregulated Internet utilities environment. To this end, the invention intends to remove heterogeneity in resource management across heterogeneous servers via the use of service units.

The administration of such resource management may be delegated to an intermediary. In particular, a mechanism is provided that allows the resources provided by media servers scattered throughout the network to suit the characteristics and needs of, or parts of, the subscriber mass.

The invention enables a remote authority to drive the administration of service units across servers based on some criteria such as, for example, the collective needs of such servers as opposed to the needs of a particular server. The remote authority may comprise a Service Control Plane (SCP) which functions as an intermediary control device between clients and servers in a distributed computer network.

As it is desirable for a meta-resource to be useful to the remote authority, the meta-resource is: (1) autonomous, (2) trusted, (3) configurable, and (4) able to offer a position of value to the remote administration. The meta-resource is to remain autonomous because, otherwise surrendering a meta-resource to the control of a remote authority would deter the proliferation of meta-resources. The present invention delivers this autonomy through three notions: service units, service signatures, and independent global vs. local resource management. A skilled artisan will thus appreciate the difference between a service commitment and a resource reservation and its role on extending autonomy over meta-resources.

Similarly, a skilled artisan will appreciate that the meta-resource needs to be trusted by the remote authority and vice-versa. Security when accessing a meta-resource is important to the content subscriber. A

mechanism is needed to enforce trust between the different parties. According to today's best practices, a key-exchange mechanism such as RSA may be used to handshake with a resource provider and authenticate the resource provider. Such mechanism is applicable to any other party. Security about the content being accessed is additionally important to the content provider. Thus, enforcement of copyrights and other forms of intellectual property protection over content is necessary. A skilled artisan will appreciate that this is a recognized need and means may be deployed to facilitate the enforcement of copyright between parties having different levels of trustiness. In particular, digital watermarking techniques may be used for safeguarding the copyrights of service objects.

It is intended that for the meta-resource to offer a position of value to the remote administration authority (200) (Figure 1), e.g., SCP, the meta-resource must react to the SCP but retain autonomous control of its resources. To this end, the invention enables the configuration of a meta-resource in terms of service units (i.e., service-oriented promises or commitments) to the SCP while the fulfillment of these commitments is left to the meta-resource. As previously mentioned in conjunction with Figure 1, meta-resources may be locally administered, each meta-resource is partitioned into two virtual media servers: a local virtual media server holding persistent content to provide services for that content provider; and, a global virtual media server, that the SCP manages, on to which content may be replicated to provide services to meet the wider needs of the framework. To this end, in the preferred embodiment of the present invention, the meta-resource is partitioned into two different and independent resource pools having separate administration. On one hand, a local resource pool is implemented by the meta-resource to provide media services to be allocated by the local meta-resource administration. On the other hand, a global resource pool is implemented by the meta-resource to provide media services to be allocated by the remote administration authority.

Figure 5 illustrates streaming of MPEG-1 content, e.g., consuming resources s1 (501), m1 (502), c1 (503) and b1 (504). Streaming of another type of content, e.g., one coded in MPEG-2, may require s2 (511), m2 (512), c2 (513) and b2 (514) resources. Thus, as a convenient means of measurement of the consumption of resources by a various classes of applications, resources are characterized as storage s and service r(m,c,b) bins. Consequently, the resource is viewed as a set of storage and service bins, which represent capabilities for provisioning services. As not all

applications are identical, it is necessary to obtain a worst resource bound for content, requiring experimental measurement of different classes of applications for various servers. However, this approach allows convenient normalization and reservation of heterogeneous server resources by various classes of applications.

As mentioned, a system administrator configures overall resources as local and global. The administrator is responsible for establishing the ratio of local to global resources for each server as well as to establishing policies relating to load limits for those resources. After configuring a partition as global storage, the global resource management takes over the control of this resource. Thus, a global storage bin represents a partition that can only be reserved by the global resource management provided by SCP. Note, that the system administrator or the server (500) itself, depending on a relevant policy, may re-claim the global resource in full or partially, by requesting its release from the SCP management. A native reservation management process (520) as shown in Figure 5, is responsible for monitoring resource consumption, and determining the server willingness to accept new requests. It is understood that, the native process may discriminate between application requests for global and local content. Additionally, a request for placement of a global replica may be declined or accepted, as controlled by internal meta-resource policies governing the autonomy of the meta-resource with respect to admission control for example, one such policy could attempt to maximize the revenue associated with global resources and thus reject replicas expected to have high cost or little revenue. Such a policy may also depend on request statistics monitored by the SCP.

Figure 6 illustrates an example scenario where a system administrator has configured all server resources onto two partitions, one local and one global. As shown in the example scenario depicted in Figure 6, two streaming applications share a transient content replica and consume one global storage bin s2 (602) and two service bins r2 (603) and r3 (604), while one application consumes a persistent content replica included in service storage bin s1 (601). Overall global storage resource consumed is thus equivalent to s2, while an overall global service resource is a sum of bins r2 and r3.

Referring back to Figure 4, there is illustrated an example use of service and storage bins and the use of global and local bins in a meta-resource as prescribed according to the present invention. As

previously mentioned, a service unit represents a service-oriented resource commitment by a meta-resource to provision a corresponding media service. Similarly, a service unit may be committed to be global or local, or both. On one hand, a local service unit is used to denote a commitment for a unit of service by a particular meta-resource where such meta-resource is to determine the binding of resources (associated with a service unit) to some service object (to be provisioned via those resources). On the other hand, a global service unit is used to denote a commitment of a unit of service by a particular meta-resource where the binding of such resources to some service object is to be determined at a later time by an external party to the meta-resource. The meta-resource may comprise a list (140) of service objects c1,..,c4 with a mapping as being locally available (L), globally available (G), or both (L,G).

For each type of service (i.e., capability) being provided by a meta-resource as indicated in a capabilities bank (130), a number of service units is pre-allocated. Thus, as shown in Figure 4, ten (10) service units "A" may be committed to a local pool, while five (5) service units "A" may be committed to the local pool. Preferably, the pre-allocation is done independently for both the global and local pools. The administration of the meta-resource defines this number (herein referred to as the capacity of a capability) according to some criteria such as expected revenue or availability. For example, a meta-resource could be made to be from 0% to a 100% global (i.e., shareable with the global meta-system), as determined by the meta-resource administration. It should be noted that resources associated with fractional service units in a meta-resource might be allocated into a third pool referred to as an overflow pool (not shown). That is, after allocating both global and local service bins according to requirements or needs for provisioning services, there may be resources leftover. These remainder resources are managed as a overflow pool (733) as shown in Figure 7. The overflow pool thus contains resources that are not being reserved for provisioning any service bins and thus, are free to be allocated by the meta-resource (server) as deemed necessary. For example, the overflow pool may be used to provide the resources needed to provide run-time resource compensation.

It is a preferred aspect of the present invention that all these three different pools are independently administered. The present invention enables administration of these resource pools in terms of the type and number of service units. The administrator of a meta-resource is thus able to enhance the value such meta-resource offers to the remote

authority by reconfiguring these resource pools to suit demand. Such administration may be based on some criteria such as the expected revenue, or some other cost metric. It is a further preferred aspect of the present invention that individual meta-resources be entitled to various degrees of greediness toward acquisition of service requests by configuring their resource pools and capabilities to suit demand and revenue expectations. Moreover, it is a preferred aspect of the present invention that such administration of the meta-resource may be done remotely. In particular, the ratios of capacities between global, local, and overflow may be set dynamically based on the needs of a remote authority, for example.

Media services on a meta-resource are modeled in terms of capabilities. Herein, a capability is used to refer to a media flow processing function which, for example, takes a stream f as input, applies a transform over the stream, and outputs a stream f'. Examples of such functions include, but are not limited to: media filters, encoders, decoders, transcoders, multiplexers, de-multiplexers, frame counters, object detectors, flow encryption, content-based filtering, content translation, etc. It is understood that such capabilities may be downloaded into meta-resources. A meta-resource however, may offer a desirable tradeoff in some resource management characteristics (e.g., the desired bandwidth at the desired geographical location) but may be lacking the necessary service management characteristics (e.g., the presence of a desired capability). Thus, according to the present invention, a meta-resource may request and download some of its capabilities from a remote capabilities bank.

To this end, a mechanism is provided to download a capability to a service insertion point meta-resource. An authority (e.g., the remote administration authority) may be used to determine the proper placement of capabilities through the collection of distributed meta-resources and where such placement could be based on some set criteria such as a cost benefit analysis with respect to bandwidth, geography, or utilization, for example. Alternately, or in addition, the placement of a capability may be auctioned as opposed to having the placement of such capability assigned by some remote administration. Last, a mechanism is provided to permit a meta-resource to receive recommendations about configuration and management of its resources and capabilities such, as for example, to suit the needs of the remote administration.

Via access controls over capabilities and service units, the resource provider is now enabled to grant or deny access to the download of capabilities as well as the administration and configuration of its resources into service units.

Figure 7 depicts the components of service unit resource management at a meta-resource. A service unit signaling adapter (700) enables the meta-resource to receive provisioning requests (705) in terms of service units. The signaling adapter (700) allows the meta-resource to communicate to external parties in terms of service units, provisioning requests, service signatures, and resource envelopes. The service unit management adapter (700) enables a remote intermediary controller to interface with resource management in this meta-resource. A service unit database (710) provides the service unit management module (SUMM) (720) with the resource envelope associated with a particular service unit. The service unit management module (720) coordinates the translation of service units to individual resource requirements and interfaces with the service unit mapper (730) to place coordinated requests as well as updates for resource allocation across one or more resources. Specifically, the coordinated services to resource mapping (730) receives a service bin specification and generates individual resource allocations based on the compensated service unit's resource envelope. Operating system interfaces provide access to individual resources such as CPU (761), memory (762), network bandwidth (763), disk bandwidth and disk storage (764). The service unit mapper module (730) additionally provides interfaces to the individual resource management interfaces. A run-time compensation module (735) computes the necessary adjustments over the resource envelope. The service unit management module (720) allows the control of resources in terms of application level units as opposed to individual resource requirements. Resources are thus allocated in terms of application requirements and optimized to local and global cost metrics as opposed to rigid resource utilization. The SUMM enables implementation of run-time adjustments to be made to the service unit envelope as well as run-time adjustments to the service unit weighting function.

It should be understood that, in addition to servicing requests, the remote authority could push (i.e., update) the definition of a particular service unit (i.e., envelopes and signatures) to a meta-resource. Clearly this is needed when a new service is being introduced, as otherwise each service unit would need to be loaded manually at each server. If a resource envelope is pushed into a meta-resource, the meta-resource updates

its service unit database (710) with the new resource envelope. A skilled artisan will appreciate that such operation is desirable when new service objects are introduced and resource envelopes have been recommended by the creator of the service object. If a provisioning request is dispatched to a meta-resource, the service unit management module (SUMM) determines whether the provisioning request can be scheduled by the meta-resource given the available resources. If a service signature is pushed into the meta-resource, the SUMM updates the corresponding service signature during the handling of future updates. If a reservation for a number of service units is requested to the meta-resource, the SUMM determines whether the reservation can be scheduled given the projected resources.

With further regard to compensation, as the possibility may exist that a resource envelope projected by a service unit may incorrectly estimate the resource requirements needed to provision the service object, existing interfaces are provided by the server operating system to permit monitoring of resource reservation exceptions. The resource management feedback unit (740) receives these exceptions and forwards these to the run-time compensation unit (735), which in turn computes the departure on the resulting resource envelope utilizing heuristics provided in a service unit heuristics database. Specifically, the resource management feedback module (740) is a software handler that maintains an association of individual resource monitors to a service unit and triggers a compensation of the resource envelope for a service unit during run-time. Once a service unit is allocated, individual resource monitors are started and associated with a common service unit. In case of an allocation exception or predicted under/over usage condition (against that predicted by the service unit's resource envelope) an exception may be fired by any one or more of these individual resource monitors. For example, if bandwidth is predicted to be low, the network I/O monitor (not shown) will signal such condition to the resource management feedback module which determines whether additional bandwidth needs to be allocated. To make such decision, the resource management feedback module (740) may rely on heuristics or policies. The service unit heuristics database (725) includes rules about how to operate over service unit as first class objects. For example, it knows/learns that for certain service units, allocating two of them at the same time really means allocating 2x the resources whereas for others it means a worst case bound of, say, 1.4x the resources. This database additionally comprises data known to artisans in adaptive resource management such as bounds over resource adjustments, periodicity of resource adjustments, relative priorities of resource adjustments, etc.

If a compensation is deemed necessary, a signal is sent to the run-time resource compensation unit (735) which may apply to one or more resources. Particularly, the run-time resource compensation unit operates on stimuli from the feedback module and knowledge from the heuristics database. The run-time compensation unit implements the adjustments or compensations and determines the order of adjustments. It also plans to prevent deadlocks and manages the overflow pool (733) which interfaces with the compensation unit to support run-time compensations. If the departure requires compensation, the compensation recommendation is forwarded to the SUMM, which first places a request for a resource envelope update from the coordinated resource management unit (730) and then determines whether the corresponding envelope needs to be updated in the service unit database to avoid similar problems in the future. A skilled artisan will appreciate that it may not always be advantageous to update the service unit database as load anomalies may require anomalous resource allocations, which may significantly exceed the requirements of average provisioning requests.

Figure 8(a) is a flow chart depicting in greater detail the process for handling a provisioning request (800). As shown in Figure 8(a), the signaling adapter receives the provisioning request and then forwards any such request to the SUMM which then interfaces to the service unit database in order to retrieve and update resource envelopes (805). At step (810), the service unit signature for the particular requested service is compared with resources at a particular server. Specifically, when a request arrives at the meta-resource, it is necessary to determine whether the request can be serviced, i.e., if the meta-resource is capable, has the resources, is willing to, and has the necessary capability. All these decisions are abstracted by the service unit. Therefore, a determination is made at step (815) as to whether a service unit in a meta-resource is present indicating that the server is capable of provisioning such unit, i.e., that the necessary resources are present. The presence of a service unit provides the ability to determine the willingness of the server in accepting a request. If the service unit is not present, the request fails and the process ends without fulfillment of the request. If the service unit is present, then at step (820) a determination is made as to whether the meta-resource is willing to accept the request, i.e., if the server is willing to provide the media service when criteria such as price, current service unit utilization, and access controls, for example, are considered. Specifically, after a request arrives to the meta-resource, the meta-resource must decide whether to service the request or not. Such decision is supported by the meta-data in the resource. For example, the

meta-resource (i.e., the server) determines whether the requests is associated with the right access controls (permissions) to use the service/storage bins being requested. Other criteria are price/cost admissibility. For example, the request may bound cost to $4.00 for example, whereas the meta-resource is willing to provide the service at $3.00. At step (825) the process will terminate if the request is not admissible, or, will continue otherwise. At step (835) any resource envelope adjustments are made and, at step (840), the adjusted service unit is allocated. For example, a service request may request a service unit (X, Y, Z) resource units of respective resources and is currently being serviced. A second request requests (X, Y, Z). For the adjustment step (835), a heuristics database look-up is performed and a determination made as to the form of the resulting resource allocation (f(X), g(Y), h(Z)) given the class of server (meta-resource). Once the resources are determined, any extra resources can be transferred to the overflow pool (e.g., for the duration associated for the provisioning of this request). This is accomplished during step (840) as well. Then, at step (850) the resource monitors are invoked by the operating system of the provisioning meta-resource (server) to monitor actual resources utilized in the provisioning of the requested service which is provided to the client as indicated at step (855). After provisioning of the service, the process ends at step (860) and returns to process more requests at step (865). Typically, the SUMM (Figure 7) renders all its comparisons and determinations based on the corresponding resource envelope associated with a particular request and then requests the coordination and allocation of the service unit. However, the coordination between the various resources associated with a particular service unit is provided by the coordinated resource management module (730). In turn, the coordinated resource management module interfaces with the resource management interfaces (750) provided by the operating system found on the meta-resource.

Figure 8(b) is a flow chart depicting in greater detail the real-time resource monitor process thread invoked at step (850) of Figure 8(a). Typically, this functionality is standard in most computer operating systems. For example, resources that may be monitored include virtual memory and page hits, stream I/O and buffer management, CPU and CPU load scheduling and priority handling, etc (See Figure 7). As shown in Figure 8(b), at step (875) the requirements departure is estimated, e.g., the number of I/O buffers needed to stream (e.g., 1MB) and bandwidth. Techniques such as optimal smoothing of recorded and live video allow estimating reliably these values and determining under/over flow

conditions. As it may be the case that the resource monitor (750) may not react on the first trigger, the monitored input data may be smoothed due to the nature of conventional operating systems. For example, an exponential smoother may be used. Trend assessment may additionally be performed as well. A critical threshold (that would be stored in the heuristics database) associated with each particular resource may be used to determine whether any departure is statistically significant, thus resulting in the generation of an exception (step 880). For example, correcting a 0.0001 departure is not a significant departure. Then, at step (880), a determination is made as to whether an exception has been detected. If an exception has not been detected, then the process returns at step (890). If an exception has been detected, then the resource allocation is compensated, as indicated at step (885) in the manner as described herein. Otherwise, the process returns at step (890).

While the present invention has been particularly shown and described with respect to illustrative and preformed embodiments thereof, it will be understood by those skilled in the art that the foregoing and other changes in form and details may be made therein without departing from the scope of the invention which should be limited only by the scope of the appended claims.

CLAIMS

1. A system for administering resources in heterogenous server devices included in a distributed computer network, said resources including media content and service resources for providing said media content to clients, said system comprising:

a) for each server device, a first resource pool comprising resources for providing media services and capable of being locally administered; and a second resource pool comprising resources for providing media services and capable of being remotely administered; each said media service having an associated homogeneous service unit representing a resource allocation commitment for provisioning said media service by a particular server device; and in which each of said service unit associated with a media service projects an envelope of resource requirements as needed for provisioning a media service; and

b) an administrator device for receiving client requests and determining a server device to provide a requested service, said administrator device accessing said homogeneous service units and comparing said homogeneous service units of associated server devices according to a predetermined criteria to determine said server device, whereby said homogeneous server units provide a basis for managing resources of heterogeneous server devices in a distributed network.

2. A system as claimed in Claim 1, in which each of said service unit associated with a media service includes an associated service signature comprising metadata for enabling access to a service unit, each said server device including means for enforcing said service signatures.

3. A system as claimed in Claim 2, in which said service signature includes metadata representing rights, privileges and characteristics of services that may use a particular service unit.

4. A system as claimed in Claim 2, in which said server device further comprises:

a) means for monitoring resource utilization when provisioning a media service; and

b) means for compensating for differences between true resource utilization when providing a media service and its resource envelope as projected in its associated service unit.

5. A system as claimed in Claim 4, in which said service signature further includes one or more run-time compensation strategies for updating a projected resource envelope associated with a service unit at different network loads.

6. A system as claimed in Claim 1, in which a projected resource envelope includes data indicating one or more resource allocations selected from the group comprising: memory usage; disk storage; disk bandwidth; CPU resources; and network bandwidth.

7. A system as claimed in Claim 6, in which one or more said resource allocations includes a time-variant profile.

8. A system as claimed in Claim 1, wherein said predetermined criteria includes a cost metric.

9. A system as claimed in Claim 1, whereby allocation of resources in said first and second resource pools is determined by a local administrator for said server device.

10. A system as claimed in Claim 9, in which each of said first and second resource pools comprises one or more service bins allocated according to service units and one or more storage bins allocated according to media content objects.

11. A system as claimed in Claim 9, further comprising a capabilities bank for downloading capabilities to said server device for configuring said service and storage bins thereof according to a determined administration.

12. A system as claimed in Claim 9, further comprising means for generating a streaming connection comprising a media service, said generating means including binding a storage bin to a service bin, the binding being evaluated at a quality of service.

13. A system as claimed in Claim 12, in which a storage bin may be bound to a service bin from a same or different resource pool.

14. A method for administering resources included in heterogeneous server devices comprising a distributed computer network, said server devices including media content and service resources for providing said media content to clients, said method comprising the steps of:

a) receiving a request for provisioning a media service to a client;

b) accessing homogeneous service units associated with the requested service, each service unit representing a resource allocation commitment for provisioning the requested media service from a particular server device and in which each of said service unit associated with a media service projects an envelope of resource requirements as needed for provisioning a media service; and

c) determining a server device having resources capable of provisioning said service according to requirements specified in its associated service unit, whereby homogeneous service units provides basis for managing resources of heterogeneous server devices in said distributed network.

15. A method as claimed in Claim 14, in which said determining step c) includes comparing different service units associated with a requested service according to a predetermined cost metric.

16. A method as claimed in Claim 14, further including step d) of allocating resources at a server device according to a service unit in order to provision a media service.

17. A method as claimed in Claim 16, in which each of said service unit associated with a media service includes an associated service signature comprising metadata used to enable access to a service unit, said accessing step b) further including the step of enforcing said service signatures at a media server device.

18. A method as claimed in Claim 16, further comprising the steps of:

e) monitoring true resource utilization at a server device when providing said media service; and

f) real-time compensating for differences between true resource utilization when providing said media service and its resource envelope as projected in its associated service unit.

19. A method as claimed in Claim 18, in which resources are allocated at a server device from a first resource pool, said method including transferring remaining resources not being used for provisioning a service in a second pool of resources.

20. The method as claimed in Claim 19, in which said compensating step f) includes allocating resources from said second pool of resources for supporting real-time compensations.

21. A method as claimed in Claim 19, in which said first resource pool comprises one or more service bins allocated according to service units and one or more storage bins allocated according to media content objects.

22. A method as claimed in Claim 21, further comprising the step of downloading capabilities from a remote capabilities bank to said server device for configuring said service and storage bins thereof according to a determined administration.

23. A method as claimed in Claim 22, in which said step of provisioning said media service further includes generating a streaming connection between a server device and a client by binding a storage bin to a service bin, the binding being evaluated at a quality of service.

(19) Canadian Intellectual Property Office
An Agency of Industry Canada

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

(11) CA 2 467 998 (13) A1

(40) 24.02.2000
(43) 24.02.2000

(12)

(21) 2 467 998

(22) 12.08.1999

(51) Int. Cl.[7]: H04L 9/28, G06F 12/14, H04L 12/16, H04L 9/30, G06F 17/60

(30) 09/133,519 US 13.08.1998
09/152,756 US 22.09.1998
09/177,096 US 22.10.1998

(60) 23.06.2004
(62) 2338414

(71) INTERNATIONAL BUSINESS MACHINES CORPORATION,
New Orchard Road, ARMONK, XX (US).

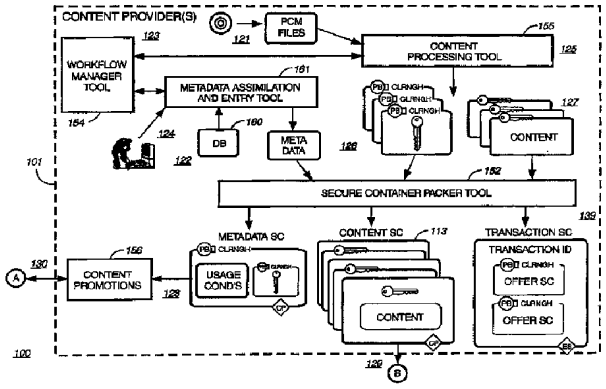
(72) HURTADO, MARCO (US).
LOTSPIECH, JEFFREY (US).
LEHMAN, CHRISTOPHER (US).
GRUSE, GEORGE GREGORY (US).
MEDINA, CESAR (US).
DORAK, JOHN (US).
DOWNS, EDGAR (US).
MILSTED, KENNETH (US).

(74) ROSEN, ARNOLD

(54) SYSTEME DE SUIVI DE L'UTILISATION DE CONTENU ELECTRONIQUE PAR L'UTILISATEUR FINAL
(54) SYSTEM FOR TRACKING END-USER ELECTRONIC CONTENT USAGE

(57)

A system for tracking usage of digital content on user devices. Electronic stores coupled to a network sell licenses to play digital content data to users. Content players, which receive from the network the licensed content data, are used to play the licensed content data. Additionally, a logging site that is coupled to the network tracks the playing of the content data. In particular, the logging site receives play information from the network, and the play information includes the number of times that the content data has been played by the associated content player. Also provided is a method for tracking usage of digital content on user devices. According to the method, a license to play digital content data is sold to a user, and the licensed content data is transmitted to a content player for the user. Further, information is transmitted to a logging site whenever the content data is played by the content player or copied from the content player to an external medium so that usage of the licensed content data can be tracked.



Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2467998 A1 2000/02/24

(21) 2 467 998

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(22) Date de dépôt/Filing Date: 1999/08/12

(41) Mise à la disp. pub./Open to Public Insp.: 2000/02/24

(62) Demande originale/Original Application: 2 338 414

(30) Priorités/Priorities: 1998/08/13 (09/133,519) US; 1998/09/22 (09/152,756) US; 1998/10/22 (09/177,096) US

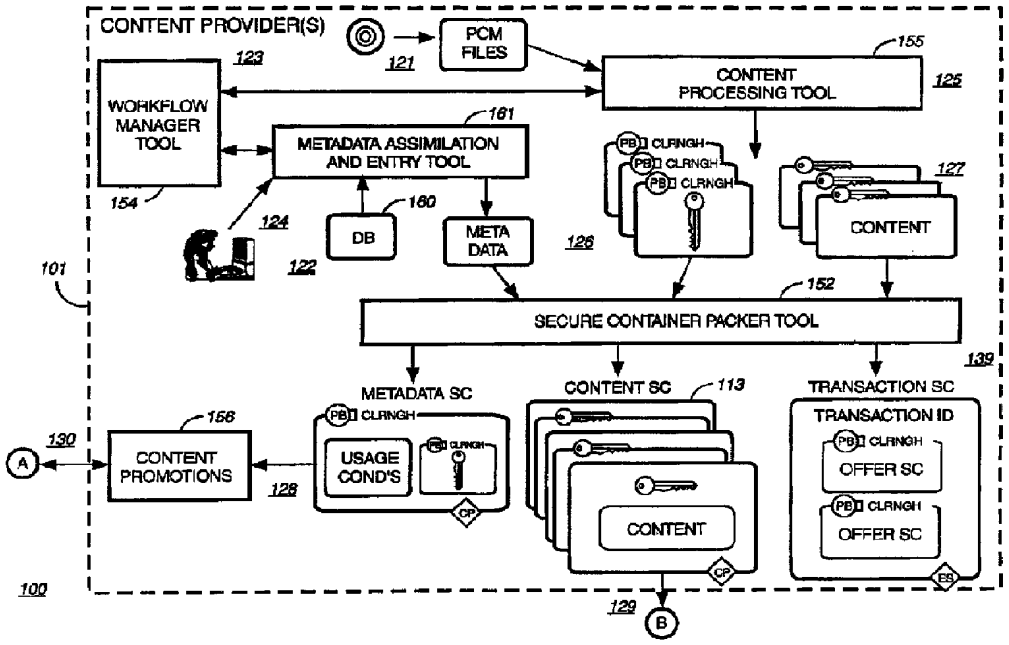
(51) Cl.Int.^{7}/Int.Cl.^{7} H04L 9/28, G06F 17/60, G06F 12/14, H04L 12/16, H04L 9/30

(71) Demandeur/Applicant:
INTERNATIONAL BUSINESS MACHINES CORPORATION, US

(72) Inventeurs/Inventors:
DORAK, JOHN, US;
DOWNS, EDGAR, US;
GRUSE, GEORGE GREGORY, US;
HURTADO, MARCO, US;
LEHMAN, CHRISTOPHER, US;
LOTSPIECH, JEFFREY, US;
MEDINA, CESAR, US;
MILSTED, KENNETH, US

(54) Titre : SYSTEME DE SUIVI DE L'UTILISATION DE CONTENU ELECTRONIQUE PAR L'UTILISATEUR FINAL

(54) Title: SYSTEM FOR TRACKING END-USER ELECTRONIC CONTENT USAGE



(57) Abrégé/Abstract:
A system for tracking usage of digital content on user devices. Electronic stores coupled to a network sell licenses to play digital content data to users. Content players, which receive from the network the licensed content data, are used to play the licensed content data. Additionally, a logging site that is coupled to the network tracks the playing of the content data. In particular, the logging site receives play information from the network, and the play information includes the number of times that the content data has been played by the associated content player. Also provided is a method for tracking usage of digital content on user devices. According to the method, a license to play digital content data is sold to a user, and the licensed content data is transmitted to a content player for the user. Further, information is transmitted to a logging site whenever the content data is played by the content player or copied from the content player to an external medium so that usage of the licensed content data can be tracked.

(74) Agent: ROSEN, ARNOLD

ABSTRACT

A system for tracking usage of digital content on user devices. Electronic stores coupled to a network sell licenses to play digital content data to users. Content players, which receive from the network the licensed content data, are used to play the licensed content data. Additionally, a logging site that is coupled to the network tracks the playing of the content data. In particular, the logging site receives play information from the network, and the play information includes the number of times that the content data has been played by the associated content player. Also provided is a method for tracking usage of digital content on user devices. According to the method, a license to play digital content data is sold to a user, and the licensed content data is transmitted to a content player for the user. Further, information is transmitted to a logging site whenever the content data is played by the content player or copied from the content player to an external medium so that usage of the licensed content data can be tracked.

SYSTEM FOR TRACKING END-USER ELECTRONIC CONTENT USAGE

BACKGROUND OF THE INVENTION

1. Field of the Invention

The invention disclosed broadly relates to the field of electronic commerce and more particularly to a system and related tools for the secure delivery and rights management of digital assets, such as print media, films, games, and music over global communications networks such as the Internet and the World Wide Web.

2. Description of the Related Art

The use of global distribution systems such as the Internet for distribution of digital assets such as music, film, computer programs, pictures, games and other content continues to grow. At the same time owners and publishers of valuable digital content have been slow to embrace the use of the Internet for distribution of digital assets for several reasons. One reason is that owners are afraid of unauthorized copying or pirating of digital content. The electronic delivery of digital content removes several barriers to pirating. One barrier that is removed with electronic distribution is the requirement of the tangible recordable medium itself (e.g., diskettes or CD ROMs). It costs money to copy digital content on to tangible media, albeit, in many cases less than a dollar for a blank tape or recordable CD. However, in the case of electronic distribution, the tangible medium is no longer needed. The cost of the tangible medium is not a factor because content is distributed electronically. A second barrier, is the format of the content itself i.e. is the content stored in an analog format versus a digital format. Content stored in an analog format, for example, a printed picture, when reproduced by photocopying, the copy is of lesser quality than the original. Each subsequent copy of a copy, sometimes called a generation, is of less quality than the original. This degradation in quality is not present when a picture is stored digitally. Each copy, and every generation of copies can be as clear and crisp as the original. The aggregate effect of perfect digital copies combined with the very low cost to distribute content electronically and to distribute content widely over the Internet makes it relatively easy to pirate and distribute unauthorized copies. With a couple of keystrokes, a pirate can send hundreds or even of thousands of perfect copies of digital

content over the Internet. Therefore a need exists to ensure the protection and security of digital assets distributed electronically.

Providers of digital content desire to establish a secure, global distribution system for digital content that protects the rights of content owners. The problems with establishing a digital content distribution system includes developing systems for digital content electronic distribution, rights management, and asset protection. Digital content that is distributed electronically includes content such as print media, films, games, programs, television, multimedia, and music.

The deployment of an electronic distribution system provides the digital content providers the ability to achieve fast settlement of payment through immediate sales reporting and electronic reconciliation as well as gain secondary sources of revenue through redistribution of content. Since the electronic digital content distribution system is not affected by physical inventory outages or returns, the digital content providers and retailers may realize reduced costs and improved margins. Digital content providers could facilitate new, or augment existing, distribution channels for better timed-release of inventory. The transactional data from the electronic distribution system could be used to obtain information regarding consumer buying patterns as well as to provide immediate feedback on electronic marketing programs and promotions. In order to meet these goals, a need exists for digital content providers to use an electronic distribution model to make digital content available to a wide range of users and businesses while ensuring protection and metering of digital assets.

Other commercially available electronic distribution systems for digital content, such as real audio, A2B from AT&T, Liquid Audio Pro from Liquid Audio Pro Corp., City Music Network from Audio Soft and others offer transmission of digital data over secured and unsecured electronic networks. The use of secured electronic networks greatly reduces the requirement of digital content providers of distributing digital to a wide audience. The use of unsecured networks such as the Internet and Web allows the digital content to arrive to an end-user securely such as through the use of encryption. However, once the encrypted digital content is de-encrypted on the end-user's machine, the digital content is readily available to the end-user for unauthorized re-distribution. Therefore a need exists for a secure digital content electronic distribution system that provides protection of digital assets and ensures that the Content Provider(s)' rights are protected even after

the digital content is delivered to consumers and businesses. A need thus exists for rights management to allow for secure delivery, licensing authorization, and control of the usage of digital assets.

Another reason owners of digital content have been slow to embrace electronic distribution is their desire to maintain and foster existing channels of distribution. Most content owners sell through retailers. In the music market these U.S. retailers include Tower Records, Peaches, Blockbuster, Circuit City and others. Many of these retailers have Web sites that allow Internet users to makes selections over the Internet and have selections mailed to the end-user. Example music Web sites include @tower, Music Boulevard and Columbia House. The use of electronic distribution can remove the ability of the retail stores from differentiating themselves from each other and differentiate themselves from the content owners, especially on the Web. Therefore a need exists to provide retailers of electronic content such as pictures, games, music, programs and videos a way to differentiate themselves from each other and the content owners when selling music through electronic distribution.

Content owners prepare their digital content for electronic distribution through distribution sites such as electronic stores. Electronic stores on the Internet, or through other online services, want to differentiate themselves from each other by their product offerings and product promotions. A traditional store, i.e. - the non-electronic, non-online analogs to electronic stores - use product promotions, product sales, product samples, liberal return policies and other promotional programs to differentiate themselves from their competitors. However, in the online world where the content providers impose usage conditions on the digital content, the ability of electronic stores to differentiate themselves may be severely limited. Moreover, even if the usage conditions can be changed, electronic stores are faced with the difficult task of processing the metadata associated with the digital content from the content providers to promote and sell products electronically. Electronic stores need to manage several requirements when processing the metadata. First, the electronic store is required to receive the metadata associated with the digital content from the content providers. Many times, parts of this metadata may be sent encrypted, so the content provider must create a mechanism to decrypt the encrypted content. Second, the electronic store may wish to preview metadata from the content provider either before the content is received from the content provider

or after the content is received by the electronic store, in order to assist with product marketing, product positioning and other promotional considerations for the content. Third, the electronic store is required to extract certain metadata used for promotional materials such as graphics and artist information. Often, this promotional material is used directly by the electronic store in its online promotions. Fourth, the electronic stores may wish to differentiate themselves from one another by modifying some of the permitted usage conditions to create different offerings of the digital content. Fifth, the electronic store may have to insert or alter certain address, such as URLs, in the metadata to direct payment reconciliation to an account reconciliation house automatically by the purchaser without the need to go through the electronic store for payment clearance. Sixth, the electronic store may need to create licenses for the permitted use of the copyrighted digital content that match usage conditions. For example, the license may grant the permission to make a limited number of copies of the digital content. A license is needed to reflect the terms and conditions of the permission granted.

In light of all these requirements, to process the metadata related to the digital content, many electronic stores write customized software programs to handle these requirements. The time, cost and testing needed to create these customized software programs can be large. Accordingly, a need exists to provide a solution to these requirements.

Still, another reason owners of digital content have been slow to embrace electronic distribution is the difficulty in preparing content for electronic distribution. Today, many providers of content have thousands or even tens of thousands of titles in their portfolio. In a music example, it is not unusual for a content owner to have a single master sound recording available on several different formats simultaneously (e.g. CD, tape and MiniDisc). In addition, a single format can have a master sound recording re-mastered or re-mixed for a specific distribution channel. As an example, the mixing for broadcast radio may be different than the mixing for a dance club sound track, which may be different than a generally available consumer CD. Inventorying and keeping track of these different mixes can be burdensome. Moreover, many owners of master recordings often times re-issue old recordings in various subsequent collections, such as "The Best Of", or in compilations for musical sound tracks to movies and other collections or compilations. As more content is offered digitally, the need to re-mix and encode the content for electronic distribution grows. Many times

providers need to use old recording formats as guides to select the correct master sound recordings and have these sound recordings reprocessed and encoded for release for electronic distribution. This may be especially true for content providers that wish to use their old formats to assist them in re-releasing the old sound recording for electronic distribution. Providers will look through databases to match up titles, artists and sound recordings to set the encoding parameters. This process of manually searching databases for recording portfolios is not without its shortcomings. One shortcoming is the need to have an operator manually search a database and set the processing parameters appropriately. Another shortcoming is the possibility of operator transcription error in selecting data from a database. Accordingly, a need exists to provide content providers a method to automatically retrieve associated data and master recordings for content such as audio.

Content owners prepare their digital content for electronic distribution through a process known as encoding. Encoding involves taking the content, digitizing it, if the content is presented in an analog format, and compressing it. The process of compressing allows the digital content to be transferred over networks and stored on recordable medium more efficiently because the amount of data transmitted or stored is reduced. However, compression is not without its shortcomings. Most compression involves the loss of some information, and is called lossy compression. Content providers must make decisions on what compression algorithm to use and the compression level required. For example, in music, the digital content or song may have very different characteristics depending on the genre of the music. The compression algorithm and compression level selected for one genre may not be the optimal choice for another genre of music. Content providers may find certain combinations of compression algorithms and compression levels work very well for one genre of music, say classical, but provide unsatisfactory results for another genre of music such as heavy metal. Moreover, audio engineers must often equalize the music, perform dynamic range adjustments and perform other preprocessing and processing settings to ensure the genre of music encoded produces the desired results. The requirement to always have to manually set these encoding parameters such as setting the equalization levels and the dynamic range settings for each digital content can be burdensome. Returning to the music example, a content provider for music with a collection covering a variety of musical genre would have to manually select for each song or set of songs to be encoded, the desired combination of encoding parameters. Accordingly, a need exists

to overcome the need for manually selection of process parameters for encoding.

The process to compress content can require a large amount of dedicated computational resources, especially for larger content items such as full-length feature movies. Providers of compression algorithms offer various tradeoffs and advantages associated with their compression techniques. These tradeoffs include: the amount of time and computational resources needed to compress the content; the amount of compression achieved from the original content; the desired bit rate for playback; the performance quality of the compressed content; and other factors. Using an encoding program which take as input a multimedia file and generate an encoded output file with no interim indication of progress or status is a problem. Moreover, in many circumstances, other programs are used to call or to manage an encoding program with no interim indication of progress. This leaves the calling application with no way to gauge the amount of content that has been encoded as a percentage of the entire selection of designated to be encoded. In circumstances where the calling program is trying to schedule several different programs to run at once this can be a problem. Furthermore, this can be especially burdensome in cases where batches of content have been selected for encoding and the content provider wants to determine the progress of the encoding process. Accordingly, a need exists to overcome these problems.

Yet, still another reason digital content providers have been slow to adopt electronic distribution for their content is lack of standards for creating digital players on end-user devices for electronically delivered content. Content providers, electronic stores, or others in the electronic distribution chain may want to offer customized players on a variety of devices such as PCS, set-top boxes, hand-held devices and more. A set of tools that can handle the decryption of the digital content in a tamper resistant environment, that is, an environment to deter the unauthorized access to the content during playing by a third party is needed. Moreover, a set of tools is needed to enable an end user to manage of a local library of digital content without allowing the end user to have access to the content for uses other than what was purchased.

Further information on the background of protecting digital content can be found from the following three sources. "Music on the Internet and the Intellectual Property Protection Problem" by Jack Lacy, James Snyder, David Maher, of AT&T Labs, Florham Park, N.J. available online URL http://www.a2bmusic.com/about/papers/musicipp.htm. Cryptographically protected container, called

DigiBox, in the article "Securing the Content, Not the Wire for Information Commerce" by Olin Sibert, David Bernstein and David Van Wie, InterTrust Technologies Corp. Sunnyvale, CA available online URL http://www.intertrust.com/architecture/stc.html. And "Cryptolope Container Technology", an IBM White Paper, available online URL http:///cyptolope.ibm.com/white.htm.

SUMMARY OF THE INVENTION

It is an object of the present invention to remove the above-mentioned drawbacks and to provide a system for tracking usage of content data. One embodiment of the present invention provides a system for tracking usage of digital content on user devices. Electronic stores coupled to a network sell licenses to play digital content data to users. Content players, which receive from the network the licensed content data, are used to play the licensed content data. Additionally, a logging site that is coupled to the network tracks the playing of the content data. In particular, the logging site receives play information from the network, and the play information includes the number of times that the content data has been played by the associated content player.

A further embodiment of the present invention provides a method for tracking usage of digital content on user devices. According to the method, a license to play digital content data is sold to a user, and the licensed content data is transmitted to a content player for the user. Further, information is transmitted to a logging site whenever the content data is played by the content player or copied from the content player to an external medium so that usage of the licensed content data can be tracked.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram illustrating an over view of a Secure Digital Content Electronic Distribution System according to the present invention.

FIG. 2 is a block diagram illustrating an example Secure Container (SC) and the associated graphical representations according to the present invention.

FIG. 3 is a block diagram illustrating an overview of the encryption process for a Secure Container (SC) according to the present invention.

FIG. 4 is a block diagram illustrating an overview of the de-encryption process for a Secure

Container (SC) according to the present invention.

FIG. 5 is a block diagram illustrating an overview of the layers for the Rights Management Architecture of the Secure Digital Content Distribution System of FIG. 1 according to the present invention.

FIG. 6 is a block diagram illustrating an overview of the Content Distribution and Licensing Control as it applies to the License Control Layer of FIG. 5.

FIG. 7 is an illustration of an example user interface for the Work Flow Manager Tool of FIG. 1 according to the present invention.

FIG. 8 is a block diagram of the major tools, components and processes of the Work Flow Manager corresponding to the user interface in FIG. 7 according to the present invention.

FIG. 9 is a block diagram illustrating the major tools, components and processes of an Electronic Digital Content Store of FIG. 1 according to the present invention.

FIG. 10 is a block diagram illustrating the major components and processes of an End-User Device(s) of FIG. 1 according to the present invention.

FIG. 11 is a flow diagram of a method to calculate an encoding rate factor for the Content Preprocessing and Compression tool of FIG. 8 according to the present invention.

FIG. 12 is a flow diagram of a method to automatically retrieve additional information for the Automatic Metadata Acquisition Tool of FIG. 8 according to the present invention.

FIG. 13 is a flow diagram of a method to automatically set the Preprocessing and Compression parameters of the Preprocessing and Compression Tool of FIG. 8 according to the present invention.

FIG. 14 is an example of user interface screens of the Player Application downloading content to a local library as described in FIG. 15 according to the present invention.

FIG. 15 is a block diagram illustrating the major components and processes of a Player Application running on End-User Device of FIG. 9 according to the present invention.

FIG. 16 is an example user interface screens of the Player Application of FIG. 15 according to the present invention.

FIG. 17 is a flow diagram of an alternate embodiment to automatically retrieve additional information for the Automatic Metadata Acquisition Tool of FIG. 8 according to the present

invention.

DETAILED DESCRIPTION OF AN EMBODIMENT

A Table of Contents is provided for this present invention to assist the reader in quickly locating different sections in this embodiment.

I. SECURE DIGITAL CONTENT ELECTRONIC DISTRIBUTION SYSTEM

A. System Overview

1. Rights Management
2. Metering
3. Open Architecture

B. System Functional Elements

1. Content Provider(s)
2. Electronic Digital Content Store(s)
3. Intermediate Market Partners
4. Clearinghouse(s)
5. End-User Device(s)
6. Transmission Infrastructures

C. System Uses

II. CRYPTOGRAPHY CONCEPTS AND THEIR APPLICATION TO THE SECURE DIGITAL CONTENT ELECTRONIC DISTRIBUTION SYSTEM

A. Symmetric Algorithms

B. Public Key Algorithms

C. Digital Signature

D. Digital Certificates

E. Guide To The SC(s) Graphical Representation

F. Example of a Secure Container Encryption

III. SECURE DIGITAL CONTENT ELECTRONIC DISTRIBUTION SYSTEM FLOW

IV. RIGHTS MANAGEMENT ARCHITECTURE MODEL

A. Architecture Layer Functions

B. Function Partitioning and Flows

1. Content Formatting Layer

2. Content Usage Control Layer

3. Content Identification Layer

4. License Control Layer

C. Content Distribution and Licensing Control

V. SECURE CONTAINER STRUCTURE

A. General Structure

B. Rights Management Language Syntax and Semantics

C. Overview of Secure Container Flow and Processing

D. Metadata Secure Container 620 Format

E. Offer Secure Container 641 Format

F. Transaction Secure Container 640 Format

G. Order Secure Container 650 Format

H. License Secure Container 660 Format

I. Content Secure Container Format

VI. SECURE CONTAINER PACKING AND UNPACKING

A. Overview

B. Bill of Materials (BOM) Part

C. Key Description Part

VII. CLEARINGHOUSE(S)

A. Overview

B. Rights Management Processing
C. Country Specific Parameters
D. Audit Logs and Tracking
E. Reporting of Results
F. Billing and Payment Verification
G. Retransmissions

VIII. CONTENT PROVIDER
A. Overview
B. Work Flow Manager
1. Products Awaiting Action/Information Process
2. New Content Request Process
3. Automatic Metadata Acquisition Process
4. Manual Metadata Entry Process
5. Usage Conditions Process
6. Supervised Release Process
7. Metadata SC(s) Creation Process
8. Watermarking Process
9. Preprocessing and Compression Process
10. Content Quality Control Process
11. Encryption Process
12. Content SC(s) Creation Process
13. Final Quality Assurance Process
14. Content Dispersement Process
15. Work Flow Rules
C. Metadata Assimilation and Entry Tool
1. Automatic Metadata Acquisition Tool
2. Manual Metadata Entry Tool
3. Usage Conditions Tool

4. Parts of the Metadata SC(s)
5. Supervised Release Tool
D. Content Processing Tool
1. Watermarking Tool
2. Preprocessing and Compression Tool
3. Content Quality Control Tool
4. Encryption Tool
E. Content SC(s) Creation Tool
F. Final Quality Assurance Tool
G. Content Dispersement Tool
H. Content Promotions Web Site
I. Content Hosting
1. Content Hosting Sites
2. Content Hosting Site(s) 111 provided by the Secure Digital Content Electronic Distribution System

IX. ELECTRONIC DIGITAL CONTENT STORE(S)
A. Overview - Support for Multiple Electronic Digital Content Store(s)
B. Point-to-Point Electronic Digital Content Distribution Service
1. Integration Requirements
2. Content Acquisition Tool
3. Transaction Processing Module
4. Notification Interface Module
5. Account Reconciliation Tool
C. Broadcast Electronic Digital Content Distribution Service

X. END-USER DEVICE(S)
A. Overview
B. Application Installation

C. Secure Container Processor
D. The Player Application
1. Overview
2. End-User Interface Components
3. Copy/Play Management Components
4. Decryption 1505, Decompression 1506 and Playback Components
5. Data Management 1502 and Library Access Components
6. Inter-application Communication Components
7. Other Miscellaneous Components
8. The Generic Player

I. SECURE DIGITAL CONTENT ELECTRONIC DISTRIBUTION SYSTEM

A. System Overview

The Secure Digital Content Electronic Distribution System is a technical platform that encompasses the technology, specifications, tools, and software needed for the secure delivery and rights management of Digital Content and digital content-related content to an end-user, client device. The End-User Device(s) include PCS, set top boxes (IRDs), and Internet appliances. These devices may copy the content to external media or portable, consumer devices as permitted by the content proprietors. The term Digital Content or simply Content, refers to information and data stored in a digital format including: pictures, movies, videos, music, programs, multimedia and games.

The technical platform specifies how Digital Content is prepared, securely distributed through point-to-point and broadcast infrastructures (such as cable, Internet, satellite, and wireless) licensed to End-User Device(s), and protected against unauthorized copying or playing. In addition, the architecture of the technical platform allows for the integration and migration of various technologies such as watermarking, compression/encoding, encryption, and other security algorithms as they evolve over time.

The base components of the Secure Digital Content Electronic Distribution System are: (1) rights management for the protection of ownership rights of the content proprietor; (2) transaction

metering for immediate and accurate compensation; and (3) an open and well-documented architecture that enables Content Provider(s) to prepare content and permit its secure delivery over multiple network infrastructures for playback on any standard compliant player.

1. Rights Management

Rights management in the Secure Digital Content Electronic Distribution System is implemented through a set of functions distributed among the operating components of the system. Its primary functions include: licensing authorization and control so that content is unlocked only by authorized intermediate or End-User(s) that have secured a license; and control and enforcement of content usage according to the conditions of purchase or license, such as permitted number of copies, number of plays, and the time interval or term the license may be valid. A secondary function of rights management is to enable a means to identify the origin of unauthorized copies of content to combat piracy.

Licensing authorization and control are implemented through the use of a Clearinghouse(s) entity and Secure Container (SC) technology. The Clearinghouse(s) provides licensing authorization by enabling intermediate or End-User(s) to unlock content after verification of a successful completion of a licensing transaction. Secure Containers are used to distribute encrypted content and information among the system components. A SC is a cryptographic carrier of information or content that uses encryption, digital signatures, and digital certificates to provide protection against unauthorized interception or modification of electronic information and content. It also allows for the verification of the authenticity and integrity of the Digital Content. The advantage of these rights management functions is that the electronic Digital Content distribution infrastructure does not have to be secure or trusted. Therefore allowing transmission over network infrastructures such as the Web and Internet. This is due to the fact that the Content is encrypted within Secure Containers and its storage and distribution are separate from the control of its unlocking and use. Only users who have decryption keys can unlock the encrypted Content, and the Clearinghouse(s) releases decryption keys only for authorized and appropriate usage requests. The Clearinghouse(s) will not clear bogus requests from unknown or unauthorized parties or requests that do not comply with the content's usage conditions as set by the content proprietors. In addition, if the SC is tampered with during its

transmission, the software in the Clearinghouse(s) determines that the Content in a SC is corrupted or falsified and repudiate the transaction.

The control of Content usage is enabled through the End-User Player Application 195 running on an End-User Device(s). The application embeds a digital code in every copy of the Content that defines the allowable number of secondary copies and play backs. Digital watermarking technology is used to generate the digital code, to keep it hidden from other End-User Player Application 195, and to make it resistant to alteration attempts. In an alternate embodiment, the digital code is just kept as part of the usage conditions associated with the Content 113. When the Digital Content 113 is accessed in a compliant End-User Device(s), the End-User Player Application 195 reads the watermark to check the use restrictions and updates the watermark as required. If the requested use of the content does not comply with the usage conditions, e.g., the number of copies has been exhausted, the End-User Device(s) will not perform the request.

Digital watermarking also provides the means to identify the origin of authorized or unauthorized copies of Content. An initial watermark in the Content is embedded by the content proprietor to identify the content proprietor, specify copyright information, define geographic distribution areas, and add other pertinent information. A second watermark is embedded in the Content at the End-User Device(s) to identify the content purchaser (or licensee) and End-User Device(s), specify the purchase or license conditions and date, and add any other pertinent information.

Since watermarks become an integral part of the Content, they are carried in the copies independent of whether the copies were authorized or not. Thus the Digital Content always contains information regarding its source and its permitted use regardless of where the content resides or where it comes from. This information may be used to combat illegal use of the Content.

2. Metering

As part of its rights management functions, the Clearinghouse(s) keeps a record of all transactions where a key exchange is cleared through the Clearinghouse(s). This record allows for the metering of licensing authorization and the original conditions of use. The transaction record can be reported to responsible parties, such as, content proprietors or Content Provider(s), retailers,

and others, on an immediate or periodic basis to facilitate electronic reconciliation of transaction payments and other uses.

3. Open Architecture

The Secure Digital Content Electronic Distribution System (System) is an open architecture with published specifications and interfaces to facilitate broad implementation and acceptance of the System in the market place while maintaining rights protection for the content proprietors. The flexibility and openness of the System architecture also enable the System to evolve over time as various technologies, transmission infrastructures, and devices are delivered to the marketplace.

The architecture is open regarding the nature of the Content and its format. Distribution of audio, programs, multimedia, video, or other types of Content is supported by the architecture. The Content could be in a native format , such as linear PCM for digital music, or a format achieved by additional preprocessing or encoding, such as filtering, compression, or pre/de-emphasis, and more. The architecture is open to various encryption and watermarking techniques. It allows for the selection of specific techniques to accommodate different Content types and formats and to allow the introduction or adoption of new technologies as they evolve. This flexibility allows Content Provider(s) to pick and evolve the technologies they use for data compression, encryption, and formatting within the Secure Digital Content Electronic Distribution System.

The architecture is also open to different distribution networks and distribution models. The architecture supports content distribution over low-speed Internet connections or high-speed satellite and cable networks and can be used with point-to-point or broadcast models. In addition, the architecture is designed so that the functions in the End-User Device(s) can be implemented on a wide variety of devices, including low cost consumer devices. This flexibility allows Content Provider(s) and retailers to offer Content to intermediate or End-User(s) through a variety of service offerings and enables the users to purchase or license Content, play it back, and record it on various compliant player devices.

B. System Functional Elements

Turning now to FIG. 1, there is shown a block diagram illustrating an overview of a Secure

Digital Content Electronic Distribution System 100 according to the present invention. The Secure Digital Content Electronic Distribution System 100 encompasses several business elements that comprise an end-to-end solution, including: Content Provider(s) 101 or the proprietors of the Digital Content, Electronic Digital Content Store(s) 103, Intermediate Market Partners (not shown), Clearinghouse(s) 105, Content Hosting Site 111, Transmission Infrastructures 107, and End-User Device(s) 109. Each of these business elements use various components of the Secure Digital Content Electronic Distribution System 100. A high level description of these business elements and system components, as they pertain specifically to electronic Content 113 distribution, follows.

1. Content Provider(s) 101

Content Provider(s) 101 or content proprietor(s) are owners of original Content 113 and/or distributors authorized to package independent Content 113 for further distribution. Content Provider(s) 101 may exploit their rights directly or license Content 113 to the Electronic Digital Content Store(s) 103, or Intermediate Market Partners (not shown), usually in return for Content usage payments related to electronic commerce revenues. Examples of Content Provider(s) 101 include Sony, Time-Warner, MTV, IBM, Microsoft, Turner, Fox and others.

Content Provider(s) 101 use tools provided as part of the Secure Digital Content Electronic Distribution System 100 in order to prepare their Content 113 and related data for distribution. A Work Flow Manager Tool 154 schedules Content 113 to be processed and tracks the Content 113 as it flows through the various steps of Content 113 preparation and packaging to maintain high quality assurance. The term metadata is used throughout this document to mean data related to the Content 113 and in this embodiment does not include the Content 113 itself. As an example, metadata for a song may be a song title or song credits but not the sound recording of the song. The Content 113 would contain the sound recording. A Metadata Assimilation and Entry Tool 161 is used to extract metadata from the Content Provider(s)' Database 160 or data provided by the Content Provider(s) in a prescribed format (for a music example the Content 113 information such as CD title, artist name, song title, CD artwork, and more) and to package it for electronic distribution. The Metadata Assimilation and Entry Tool 161 is also used to enter the Usage Conditions for the Content 113. The data in Usage Conditions can include copy restriction rules, the wholesale price, and any

business rules deemed necessary. A Watermarking Tool is used to hide data in the Content 113 that identifies the content owner, the processing date, and other relevant data. For an embodiment where the Content 113 is audio, an audio preprocessor tool is used to adjust the dynamics and/or equalize the Content 113 or other audio for optimum compression quality, compress the Content 113 to the desired compression levels, and encrypt the Content 113. These can be adapted to follow technical advances in digital content compression/encoding, encryption, and formatting methods, allowing the Content Provider(s) 101 to utilize best tools as they evolve over time in the marketplace.

The encrypted Content 113, digital content-related data or metadata, and encrypted keys are packed in SCs (described below) by the SC Packer Tool and stored in a content hosting site and/or promotional web site for electronic distribution. The content hosting site can reside at the Content Provider(s) 101 or in multiple locations, including Electronic Digital Content Store(s) 103 and Intermediate Market Partners (not shown) facilities. Since both the Content 113 and the Keys (described below) are encrypted and packed in SCs, Electronic Digital Content Store(s) 103 or any other hosting agent can not directly access decrypted Content 113 without clearance from the Clearinghouse(s) and notification to the Content Provider(s) 101.

2. Electronic Digital Content Store(s) 103

Electronic Digital Content Store(s) 103 are the entities who market the Content 113 through a wide variety of services or applications, such as Content 113 theme programming or electronic merchandising of Content 113. Electronic Digital Content Store(s) 103 manage the design, development, business operations, settlements, merchandising, marketing, and sales of their services. Example online Electronic Digital Content Store(s) 103 are Web sites that provide electronic downloads of software.

Within their services, Electronic Digital Content Store(s) 103 implement certain functions of the Secure Digital Content Electronic Distribution System 100. Electronic Digital Content Store(s) 103 aggregate information from the Content Provider(s) 101, pack content and metadata in additional SCs, and deliver those SCs to consumers or businesses as part of a service or application. Electronic Digital Content Store(s) 103 use tools provided by the Secure Digital Content Electronic Distribution System 100 to assist with: metadata extraction, secondary usage conditions, SC

packaging, and tracking of electronic content transactions. The secondary usage conditions data can include retail business offers such as Content 113 purchase price, pay-per-listen price, copy authorization and target device types, or timed-availability restrictions.

Once an Electronic Digital Content Store(s) 103 completes a valid request for electronic Content 113 from an End-User(s), the Electronic Digital Content Store(s) 103 is responsible for authorizing the Clearinghouse(s) 105 to release the decryption key for the Content 113 to the customer. The Electronic Digital Content Store(s) also authorizes the download of the SC containing the Content 113. The Electronic Digital Content Store(s) may elect to host the SCs containing the Digital Content at its local site and/or utilize the hosting and distribution facilities of another Content hosting site.

The Electronic Digital Content Store(s) can provide customer service for any questions or problems that an End-User(s) may have using the Secure Digital Content Electronic Distribution System 100, or the Electronic Digital Content Store(s) 103 may contract their customer service support to the Clearinghouse(s) 105.

3. Intermediate Market Partners (not shown)

In an alternate embodiment, the Secure Digital Content Electronic Distribution System 100 can be used to provide Content 113 securely to other businesses called Intermediate Market Partners. These partners may include digital content-related companies offering a non-electronic service, such as televisions stations or video clubs, radio stations or record clubs, that distribute Content 113. These Partners may also include other trusted parties who handle material as part of making or marketing sound recordings, such as record studios, replicators, and producers. These Intermediate Market Partners requires clearance from the Clearinghouse(s) 105 in order to decrypt the Content 113.

4. Clearinghouse(s) 105

The Clearinghouse(s) 105 provides the licensing authorization and record keeping for all transactions that relate to the sale and/or permitted use of the Content 113 encrypted in a SC. When the Clearinghouse(s) 105 receives a request for a decryption key for the Content 113 from an

intermediate or End-User(s), the Clearinghouse(s) 105 validates the integrity and authenticity of the information in the request; verifies that the request was authorized by an Electronic Digital Content Store(s) or Content Provider(s) 101; and verifies that the requested usage complies with the content Usage Conditions as defined by the Content Provider(s) 101. Once these verifications are satisfied, the Clearinghouse(s) 105 sends the decryption key for the Content 113 to the requesting End-User(s) packed in a License SC. The key is encrypted in a manner so that only the authorized user can retrieve it. If the End-User's request is not verifiable, complete, or authorized, the Clearinghouse(s) 105 repudiates the request for the decryption key.

The Clearinghouse(s) 105 keeps a record of all transactions and can report them to responsible parties, such as Electronic Digital Content Store(s) 103 and Content Provider(s) 101, on an immediate, periodic, or restricted basis. This reporting is a means by which Content Provider(s) 101 can be informed of the sale of Content 113 and the Electronic Digital Content Store(s) 103 can obtain an audit trail of electronic delivery to their customers. The Clearinghouse(s) 105 can also notify the Content Provider(s) 101 and/or Electronic Digital Content Store(s) 103 if it detects that information in a SC has been compromised or does not comply with the Content's Usage Conditions. The transaction recording and repository capabilities of the Clearinghouse(s) 105 database is structured for data mining and report generation.

In another embodiment, the Clearinghouse(s) 105 can provide customer support and exception processing for transactions such as refunds, transmission failures, and purchase disputes. The Clearinghouse(s) 105 can be operated as an independent entity, providing a trusted custodian for rights management and metering. It provides billing and settlement as required. Examples of electronic Clearinghouse(s) include Secure-Bank.com and Secure Electronic Transaction (SET) from Visa/Mastercard. In one embodiment, the Clearinghouse(s) 105 are Web sites accessible to the End-User Device(s) 109. In another embodiment, the Clearinghouse(s) 105 is part of the Electronic Digital Content Store(s) 103.

5. End-User Device(s) 109

The End-User Device(s) 109 can be any player device that contains an End-User Player Application 195 (described later) compliant with the Secure Digital Content Electronic Distribution

System 100 specifications. These devices may include PCS, set top boxes (IRDs), and Internet appliances. The End-User Player Application 195 could be implemented in software and/or consumer electronics hardware. In addition to performing play, record, and library management functions, the End-User Player Application 195 performs SC processing to enable rights management in the End-User Device(s) 109. The End-User Device(s) 109 manages the download and storage of the SCs containing the Digital Content; requests and manages receipt of the encrypted Digital Content keys from the Clearinghouse(s) 105; processes the watermark(s) every time the Digital Content is copied or played; manages the number of copies made (or deletion of the copy) in accordance with the Digital Content's Usage Conditions; and performs the copy to an external media or portable consumer device if permitted. The portable consumer device can perform a subset of the End-User Player Application 195 functions in order to process the content's Usage Conditions embedded in the watermark. The terms End-User(s) and End-User Player Application 195 are used throughout this to mean through the use or running-on an End-User Device(s) 109.

6. Transmission Infrastructures 107

The Secure Digital Content Electronic Distribution System 100 is independent of the transmission network connecting the Electronic Digital Content Store(s) 103 and End-User Device(s) 109. It supports both point-to-point such as the Internet and broadcast distribution models such as digital broadcast television.

Even though the same tools and applications are used to acquire, package, and track Content 113 transactions over various Transmission Infrastructures 107, the presentation and method in which services are delivered to the customer may vary depending on the infrastructure and distribution model selected. The quality of the Content 113 being transferred may also vary since high bandwidth infrastructures can deliver high-quality digital content at more acceptable response times than lower bandwidth infrastructures. A service application designed for a point-to-point distribution model can be adapted to support a broadcast distribution model as well.

C. System Uses

The Secure Digital Content Electronic Distribution System 100 enables the secure delivery

of high-quality, electronic copies of Content 113 to End-User Device(s) 109, whether consumer or business, and to regulate and track usage of the Content 113.

The Secure Digital Content Electronic Distribution System 100 could be deployed in a variety of consumer and business-to-business services using both new and existing distribution channels. Each particular service could use a different financial model that can be enforced through the rights management features of the Secure Digital Content Electronic Distribution System 100. Models such as wholesale or retail purchase, pay-per-listen usage, subscription services, copy/no-copy restrictions, or redistribution could be implemented through the rights management of the Clearinghouse(s) 105 and the End-User Player Application 195 copy protection features.

The Secure Digital Content Electronic Distribution System 100 allows Electronic Digital Content Store(s) 103 and Intermediate Market Partners a great deal of flexibility in creating services that sell Content 113. At the same time it provides Content Provider(s) 101 a level of assurance that their digital assets are protected and metered so that they can receive appropriate compensation for the licensing of Content 113.

II. CRYPTOGRAPHY CONCEPTS AND THEIR APPLICATION TO THE SECURE DIGITAL CONTENT ELECTRONIC DISTRIBUTION SYSTEM

License Control in the Secure Digital Content Electronic Distribution System 100 is based on the use of cryptography. This section introduces basic cryptography technologies of the present invention. The use of public key encryption, symmetric key encryption, digital signatures, digital watermarks and digital certificates is known.

A. Symmetric Algorithms

In the Secure Digital Content Electronic Distribution System 100 the Content Provider(s) 101 encrypts the content using symmetric algorithms. They are called symmetric algorithms because the same key is used to encrypt and decrypt data. The data sender and the message recipient must share the key. The shared key is referred to here as the symmetric key. The Secure Digital Content Electronic Distribution System 100 architecture is independent of the specific symmetric algorithm selected for a particular implementation.

Common symmetric algorithms are DES, RC2 and RC4. Both DES and RC2 are block cipher. A block cipher encrypts the data using a block of data bits at a time. DES is an official US government encryption standard, has a 64-bit block size, and uses a 56-bit key. Triple-DES is commonly used to increase the security achieved with simple DES. RSA Data Security designed RC2. RC2 uses a variable-key-size cipher and has a block size of 64 bits. RC4, also designed by RSA Data Security, is a variable-key-size stream cipher. A stream cipher operates on a single data bit at a time. RSA Data Security claims that eight to sixteen machine operations are required for RC4 per output byte.

IBM designed a fast algorithm called SEAL. SEAL is a stream algorithm that uses a variable-length key and that has been optimized for 32-bit processors. SEAL requires about five elementary machine instructions per data byte. A 50 MHZ, 486-based computer runs the SEAL code at 7.2 megabytes/second if the 160-bit key used has already been preprocessed into internal tables.

Microsoft reports results of encryption performance benchmark in its Overview of CryptoAPI document. These results were obtained by an application using Microsoft's CryptoAPI, running on a 120-MHZ, Pentium-based computer with Windows NT 4.0.

Cipher	Key Size	Key Setup Time	Encryption Speed
DES	56	460	1,138,519
RC2	40	40	286,888
RC4	40	151	2,377,723

B. Public Key Algorithms

In the Secure Digital Content Electronic Distribution System 100, symmetric keys and other small data pieces are encrypted using public keys. Public key algorithms use two keys. The two keys are mathematically related so that data encrypted with one key can only be decrypted with the other key. The owner of the keys keeps one key private (private key) and publicly distributes the second key (public key).

To secure the transmission of a confidential message using a public key algorithm, one must use the recipient's public key to encrypt the message. Only the recipient, who has the associated

private key, can decrypt the message. Public key algorithms are also used to generate digital signatures. The private key is used for that purpose. The following section provides information on digital signatures.

The most common used public-key algorithm is the RSA public-key cipher. It has become the de-facto public key standard in the industry. Other algorithms that also work well for encryption and digital signatures are ElGamal and Rabin. RSA is a variable-key length cipher.

Symmetric key algorithms are much faster than the public key algorithms. In software, DES is generally at least 100 times as fast as RSA. Because of this, RSA is not used to encrypt bulk data. RSA Data Security reports that on a 90 MHZ Pentium machine, RSA Data Security's toolkit BSAFE 3.0 has a throughput for private-key operations (encryption or decryption, using the private key) of 21.6 kilobits/second with a 512-bit modulus and 7.4 kilobits/second with a 1024-bit modulus.

C. Digital Signature

In the Secure Digital Content Electronic Distribution System 100, the issuer of SC(s) protects the integrity of SC(s) by digitally signing it. In general, to create a digital signature of a message, a message owner first computes the message digest (defined below) and then encrypt the message digest using the owner's private key. The message is distributed with its signature. Any recipient of the message can verify the digital signature first by decrypting the signature using the public key of the message owner to recover the message digest. Then, the recipient computes the digest of the received message and compares it with the recovered one. If the message has not being altered during distribution, the calculated digest and recovered digest must be equal.

In the Secure Digital Content Electronic Distribution System 100, since SC(s) contain several data parts, a digest is calculated for each part and a summary digest is calculated for the concatenated part digests. The summary digest is encrypted using the private key of the issuer of the SC(s). The encrypted summary digest is the issuer's digital signature for the SC(s). The part digests and the digital signature are included in the body of the SC(s). The recipients of SC(s) can verify the integrity of the SC(s) and its parts by means of the received digital signature and part digests.

A one-way hash algorithm is used to calculate a message digest. A hash algorithm takes a variable-length-input message and converts it into a fixed length string, the message digest. A

one-way hash algorithm operates only in one direction. That is, it is easy to calculate the digest for an input message, but it is very difficult (computationally infeasible) to generate the input message from its digest. Because of the properties of the one-way hash functions, one can think of a message digest as a fingerprint of the message.

The more common one-way hash functions are MD5 from RSA Data Security and SHA designed by the US National Institute of Technology and Standards (NITS).

D. Digital Certificates

A digital certificate is used to authenticate or verify the identity of a person or entity that has sent a digitally signed message. A certificate is a digital document issued by a certification authority that binds a public key to a person or entity. The certificate includes the public key, the name of the person or entity, an expiration date, the name of the certification authority, and other information. The certificate also contains the digital signature of the certification authority.

When an entity (or person) sends a message signed with its private key and accompanied with its digital certificate, the recipient of the message uses the entity's name from the certificate to decide whether or not to accept the message.

In the Secure Digital Content Electronic Distribution System 100, every SC(s), except those issued by the End-User Device(s) 109, includes the certificate of the creator of the SC(s). The End-User Device(s) 109 do not need to include certificates in their SC(s) because many End-User(s) do not bother to acquire a certificate or have certificates issued by non bona-fide Certification Authorities. In the Secure Digital Content Electronic Distribution System 100, the Clearinghouse(s) 105 has the option of issuing certificates to the Electronic Digital Content Store(s) 103. This allows the End-User Device(s) 109 to independently verify that the Electronic Digital Content Store(s) 103 have been authorized by the Secure Digital Content Electronic Distribution System 100.

E. Guide To The SC(s) Graphical Representation

This document uses a drawing to graphically represent SC(s) that shows encrypted parts, non-encrypted parts, the encryption keys, and certificates. Referring now to FIG. 2 is an example drawing of SC(s) 200. The following symbols are used in the SC(s) figures. Key 201 is a public or

private key. The teeth of the key e.g. CLRNGH for Clearinghouse indicate the key owner. PB inside the handle indicates that it is a public key thus key 201 is a Clearinghouse public key. PV inside the handle indicates that it is a private key. Diamond shape is an End-User Digital Signature 202. The initials indicate which private key was used to create the signature thus in EU is the End-User(s) digital signature from table below. Symmetric key 203 is used to encrypt content. An encrypted symmetric key object 204 comprising a symmetric key 203 encrypted with a PB of CLRNGH. The key on the top border of the rectangle is the key used in the encryption of the object. The symbol or text inside the rectangle indicates the encrypted object (a symmetric key in this case). Another encrypted object, in this example a Transaction ID encrypted object 205 is shown. And Usage Conditions 206 for content licensing management as described below. The SC(s) 200 comprises Usage Conditions 206, Transaction ID encrypted object 205, an Application ID encrypted object 207, and encrypted symmetric key object 204, all signed with an End-User Digital Signature 202.

The table below shows the initials that identify the signer of SC(s).

Initial	Component
CP	Content Provider(s) 101
MS	Electronic Digital Content Store(s) 103
HS	Content Hosting Site(s) 111
EU	End-User Device(s) 109
CH	Clearinghouse(s) 105
CA	certification authority(ies) (not shown)

F. Example of a Secure Container Encryption

The tables and diagrams below provide an overview of the encryption and decryption process used to create and recover information from SC(s). The SC(s) that is created and decrypted in this process overview is a general SC(s). It does not represent any of the specific SC(s) types used for rights management in the Secure Digital Content Electronic Distribution System 100. The process consists of the steps described in FIG. 3 for encryption process.

Process Flow for Encryption Process of FIG. 3

Step	Process
301	Sender generates a random symmetric key and uses it to encrypt the content.
302	Sender runs the encrypted content through a hash algorithm to produce the content digest.
303	Sender encrypts the symmetric key using the recipient's public key. PB RECPNT refers to the recipient's public key.
304	Sender runs the encrypted symmetric key through the same hash algorithm used in step 2 to produce the symmetric key digest.
305	Sender runs the concatenation of the content digest and symmetric key digest through the same hash algorithm used in step 2 to produce the SC(s) digest.
306	Sender encrypts the SC(s) digest with the sender's private key to produce the digital signature for the SC(s). PV SENDER refers to the sender's private key.
307B	Sender creates a SC(s) file that includes the encrypted content, encrypted symmetric key, content digest, symmetric key digest, sender's certificate, and SC(s) signature.
307A	Sender must have obtained the certificate from a certification authority prior to initiating secure communications. The certification authority includes in the certificate the sender's public key, the sender's name and signs it. PV CAUTHR refers to the certifications authority's private key. Sender transmits the SC(s) to the recipient.

Process Flow for Decryption Process of FIG. 4

Step	Process
408	Recipient receives the SC(s) and separates its parts.
409	Recipient verifies the digital signature in the sender's certificate by decrypting it with the public key of the certification authority. If the certificate's digital signature is valid, recipient acquires the sender's public key from the certificate.
410	Recipient decrypts the SC(s) digital signature using the sender's public key. This recovers the SC(s) digest. PB SENDER refers to the sender's public key.
411	Recipient runs the concatenation of the received content digest and encrypted key digest through the same hash algorithm used by the sender to compute the SC(s) digest.
412	Recipient compares the computed SC(s) digest with the one recovered from the sender's

digital signature. If they are the same, recipient confirms that the received digests have not been altered and continues with the decryption process. If they are not the same, recipient discards the SC(s) and notifies the sender.

413 Recipient runs the encrypted symmetric key through the same hash algorithm used in step 411 to compute the symmetric key digest.

414 Recipient compares the computed symmetric key digest with the one received in the SC(s). If it is the same, recipient knows that the encrypted symmetric key has not been altered. Recipient continues with the decryption process. If not valid, recipient discards the SC(s) and notifies the sender.

415 Recipient runs the encrypted content through the same hash algorithm used in step 411 to compute the content digest.

416 Recipient compares the computed content digest with the one received in the SC(s). If it is the same, recipient knows that the encrypted content has not been altered. Recipient then continues with the decryption process. If not valid, recipient discards the SC(s) and notifies the sender.

417 Recipient decrypts the encrypted symmetric key using the recipient's private key. This recovers the symmetric key. PV RECPNT refers to the recipient's private key.

418 Recipient uses the symmetric key to decrypt the encrypted content. This recovers the content.

III. SECURE DIGITAL CONTENT ELECTRONIC DISTRIBUTION SYSTEM FLOW

The Secure Electronic Digital Content Distribution System 100, consists of several components that are used by the different participants of the system. These participants include the Content Provider(s) 101, Electronic Digital Content Store(s) 103, End-User(s) via End-User Device(s) 109 and the Clearinghouse(s) 105. A high level system flow is used as an overview of the Secure Digital Content Electronic Distribution System 100. This flow outlined below tracks Content as it flows throughout the System 100. Additionally it outlines the steps used by the participants to conduct the transactions for the purchase, unlocking and use of the Content 113. Some of the assumptions made in the system flow include:

- This is a system flow for a Digital Content service (Point-to-Point Interface to a PC).
- Content Provider(s) 101 submits audio Digital Content in PCM uncompressed format (as a music audio example).
- Content Provider(s) 101 has metadata in an ODBC compliant database or Content Provider(s) 101 will enter the data directly into the Content Information Processing Subsystem, or will have provided data in prescribed ASCII file format(s).
- Financial settlement is done by the Electronic Digital Content Store(s).
- Content 113 is hosted at a single Content Hosting Site(s) 111.

It should be understood by those skilled in the art that these assumptions can be altered to accommodate the exact nature of the Digital Content e.g. music, video and program and electronic distribution systems broadcast.

The following process flow in illustrated in FIG. 1.

Step	Process
121	A uncompressed PCM audio file is provided as Content 113 by the Content Provider(s) 101. Its filename is input into the Work Flow Manager 154 Tool along with the Content Provider(s)' 101 unique identifier for the Content 113.
122	Metadata is captured from the Content Provider(s)' Database 160 by the Content Information Processing Subsystem using the Content Provider(s)' 101 unique identifier for the Content 113 and information provided by the Database Mapping Template.
123	The Work Flow Manager Tool 154 is used to direct the content flow through the acquisition and preparation process at the Content Provider(s) 101. It can also be used to track the status of any piece of content in the system at any time.
124	The Usage Conditions for the Content 113 are entered into the Content Information Processing Subsystem, this can be done either manually or automatically. This data includes copy restriction rules and any other business rules deemed necessary. All of the metadata entry can occur in parallel with the Audio Processing for the data.
125	The Watermarking Tool is used to hide data in the Content 113 that the Content Provider(s)

101 deems necessary to identify the content. This could include when it was captured, where it came from (this Content Provider(s) 101), or any other information specified by the Content Provider(s) 101.

- The Content Processing Tool 125 performs equalization, dynamics adjustments and re-sampling to the Content 113 as necessary for the different compression levels supported.
- The Content 113 is compressed using the Content Processing Tool 125 to the desired compression levels. The Content 113 can then be played back to verify that the compression produces the required level of Content 113 quality. If necessary the equalization, dynamics adjustments, compression and playback quality checks can be performed as many times as desired.
- The Content 113 and a subset of its metadata is encrypted with a Symmetric Key by the SC Packer. This tool then encrypts the key using the Public Key of the Clearinghouse(s) 105 to produce an Encrypted Symmetric Key. This key can be transmitted anywhere without comprising the security of the Content 113 since the only entity that can decrypt it is the Clearinghouse(s) 105.

126 The Encrypted Symmetric Key, metadata and other information about the Content 113 is then packed into a Metadata SC by the SC Packer Tool 152.

127 The encrypted Content 113 and metadata are then packed into a Content SC. At this point the processing on the Content 113 and metadata is complete.

128 The Metadata SC(s) is then sent to the Content Promotions Web Site 156 using the Content Disbursement Tool (not shown).

129 The Content Disbursement Tool sends the Content SC(s) to the Content Hosting Site(s) 111. The Content Hosting Site(s) can reside at the Content Provider(s) 101, the Clearinghouse(s) 105 or a special location dedicated for Content Hosting. The URL for this site is part of the metadata that was added to the Metadata SC.

130 The Content Promotions Web Site 156 notifies Electronic Digital Content Store(s) 103 of new Content 113 that is added to the System 100.

131 Using the Content Acquisition Tool, Electronic Digital Content Store(s) 103 then download

the Metadata SCs that correspond to the Content 113 they wish to sell.

132 The Electronic Digital Content Store(s) 103 will use the Content Acquisition Tool to pull out any data from the Metadata SC(s) that they want to use to promote the Content 113 on their Web Site. Access to portions of this metadata can be secured and charged for if desired.

133 The Usage Conditions for the Content 113, specific to this Electronic Digital Content Store(s) 103, are entered using the Content Acquisition Tool. These Usage Conditions include the retail prices and copy/play restrictions for the different compression levels of the Content 113.

134 The Electronic Digital Content Store(s) 103 specific Usage Conditions and the original Metadata SC(s) are packed into an Offer SC by the SC Packer Tool.

135 After the Electronic Digital Content Store(s) 103 Web Site is updated, the Content 113 is available to End-User(s) surfing the Web.

136 When an End-User(s) finds Content 113 that they want to buy, they click on a content icon, such as a music icon, and the item is added to his/her shopping cart which is maintained by the Electronic Digital Content Store(s) 103. When the End-User(s) completes shopping they submit the purchase request to the Electronic Digital Content Store(s) 103 for processing.

137 The Electronic Digital Content Store(s) 103 then interacts with credit card clearing organizations to place a hold on the funds in the same way they do business today.

138 Once the Electronic Digital Content Store(s) 103 receives the credit card authorization number back from the credit card clearing organization, it stores this into a database and invokes the SC Packer Tool to build a Transaction SC. This Transaction SC includes all of the Offer SCs for the Content 113 that the End-User(s) has purchased, a Transaction ID that can be tracked back to the Electronic Digital Content Store(s) 103, information that identifies the End-User(s), compression levels, Usage Conditions and the price list for the songs purchased.

139 This Transaction SC is then transmitted to the End-User Device(s) 109.

140 When the Transaction SC arrives on the End-User Device(s) 109, it kicks off the End-User Player Application 195 which opens the Transaction SC and acknowledges the End-User's purchase. The End-User Player Application 195 then opens the individual Offer SCs and in

an alternate embodiment, may inform the user with an estimate of the download time. It then asks the user to specify when they want to download the Content 113.

141 Based on the time the End-User(s) requested the download, the End-User Player Application 195 will wake up and initiate the start of the download process by building a Order SC that contains among other things the Encrypted Symmetric Key for the Content 113, the Transaction ID, and End-User(s) information.

142 This Order SC is then sent to the Clearinghouse(s) 105 for processing.

143 The Clearinghouse(s) 105 receives the Order SC, opens it and verifies that none of the data has been tampered with. The Clearinghouse(s) 105 validates the Usage Conditions purchased by the End-User(s). These Usage Conditions must comply with those specified by the Content Provider(s) 101. This information is logged in a database.

144 Once all the checks are complete, the Encrypted Symmetric Key is decrypted using the private key of the Clearinghouse(s) 105. The Symmetric Key is then encrypted using the public key of the End-User(s). This new Encrypted Symmetric Key is then packaged into a License SC by the SC Packer.

145 The License SC is then transmitted to the End-User(s).

146 When the License SC is received at the End-User Device(s) 109 it is stored in memory until the Content SC is downloaded.

147 The End-User Device(s) 109 request from the Content Hosting Facility 111, sending the corresponding License SC for the purchased Content 113.

148 Content 113 is sent to the End-User Device(s) 109. Upon the receipt the Content 113 is de-encrypted by the End-User Device(s) 109 using the Symmetric Key.

IV. RIGHTS MANAGEMENT ARCHITECTURE MODEL

A. Architecture Layer Functions

FIG. 5 is a block diagram of the Rights Management Architecture of the Secure Digital Content Electronic Distribution System 100. Architecturally, four layers represent the Secure Digital Content Electronic Distribution System 100: the License Control Layer 501, the Content Identification Layer 503, Content Usage Control Layer 505, and the Content Formatting Layer 507. The overall functional objective of each layer and the individual key functions for each layer are described in this section. The functions in each of the layers are fairly independent of the functions in the other layers. Within broad limitations, functions in a layer can be substituted with similar functions without affecting the functionality of the other layers. Obviously, it is required that the output from one layer satisfies format and semantics acceptable to the adjacent layer.

The License Control Layer 501 ensures that:

- the Digital Content is protected during distribution against illegal interception and tampering;

- the Content 113 originates from a rightful content owner and is distributed by a licensed distributor, e.g. Electronic Digital Content Store(s) 103;
- the Digital Content purchaser has a properly licensed application;
- the distributor is paid by the purchaser before a copy of the Content 113 is made available to the purchaser or End-User(s); and
- a record of the transaction is kept for reporting purposes.

The Content Identification Layer 503 allows for the verification of the copyright and the identity of the content purchaser. The content's copyright information and identity of the content purchaser enables the source tracking of any, authorized or not, copy of the Content 113. Thus, the Content Identification Layer 503 provides a means to combat piracy.

The Content Usage Control Layer 505 ensures that the copy of the Content 113 is used in the purchaser's device according to the Store Usage Conditions 519. The Store Usage Conditions 519 may specify the number of plays and local copies allowed for the Content 113, and whether or not the Content 113 may be recorded to an external portable device. The functions in the Content Usage Control Layer 505 keep track of the content's copy/play usage and update the copy/play status.

The Content Formatting Layer 507 allows for the format conversion of the Content 113 from its native representation in the content owner's facilities into a form that is consistent with the service features and distribution means of the Secure Digital Content Electronic Distribution System 100. The conversion processing may include compression encoding and its associated preprocessing, such as frequency equalization and amplitude dynamic adjustment. For Content 113 which is audio, at the purchaser's side, the received Content 113 also needs to be processed to achieve a format appropriate for playback or transfer to a portable device.

B. Function Partitioning and Flows

The Rights Management Architectural Model is shown in FIG. 5 and this illustrates the mapping of the architectural layers to the operating components making up the Secure Digital Content Electronic Distribution System 100 and the key functions in each layer.

1. Content Formatting Layer 507

The general functions associated with the Content Formatting Layer 507 are Content Preprocessing 502 and Compression 511 at the Content Provider(s) 101, and Content De-scrambling 513 and Decompression 515 at the End-User Device(s) 109. The need for preprocessing and the examples of specific functions were mentioned above. Content Compression 511 is used to reduce the file size of the Content 113 and its transmission time. Any compression algorithm appropriate for the type of Content 113 and transmission medium can be used in the Secure Digital Content Electronic Distribution System 100. For music, MPEG ½/4, Dolby AC-2 and AC-3, Sony Adaptive Transform Coding (ATRAC), and low-bit rate algorithms are some of the typically used compression algorithms. The Content 113 is stored in the End-User Device(s) 109 in compressed form to reduce the storage size requirement. It is decompressed during active playback. De-scrambling is also performed during active playback. The purpose and type of scrambling will be described later during the discussion of the Content Usage Control Layer 505.

2. Content Usage Control Layer 505

The Content Usage Control Layer 505 permits the specification and enforcement of the conditions or restrictions imposed on the use of Content 113 use at the End-User Device(s) 109. The conditions may specify the number of plays allowed for the Content 113, whether or not a secondary copy of the Content 113 is allowed, the number of secondary copies, and whether or not the Content 113 may be copied to an external portable device. The Content Provider(s) 101 sets the allowable Usage Conditions 517 and transmits them to the Electronic Digital Content Store(s) 103 in a SC (see the License Control Layer 501 section). The Electronic Digital Content Store(s) 103 can add to or narrow the Usage Conditions 517 as long as it doesn't invalidate the original conditions set by the Content Provider(s) 101. The Electronic Digital Content Store(s) 103 then transmits all Store Usage Conditions 519 (in a SC) to the End-User Device(s) 109 and the Clearinghouse(s) 105. The Clearinghouse(s) 105 perform Usage Conditions Validation 521 before authorizing the Content 113 release to an End-User Device(s) 109.

The enforcement of the content Usage Conditions 517 is performed by the Content Usage Control Layer 505 in the End-User Device(s) 109. First, upon reception of the Content 113 copy

from the Content Identification Layer 503 in the End-User Device(s) 109 marks the Content 113 with a Copy/Play Code 523 representing the initial copy/play permission. Second, the Player Application 195 cryptographically scrambles the Content 113 before storing it 525 in the End-User Device(s) 109. The Player Application 195 generates a scrambling key for each Content item, and the key is encrypted and hidden in the End-User Device(s) 109. Then, every time the End-User Device(s) 109 accesses the Content 113 for copy or play, the End-User Device(s) 109 verifies the copy/play code before allowing the de-scrambling of the Content 113 and the execution of the play or copy. The End-User Device(s) 109 also appropriately updates the copy/play code in the original copy of the Content 113 and on any new secondary copy. The copy/play coding is performed on Content 113 that has been compressed. That is, there is no need to decompress the Content 113 before the embedding of the copy/play code.

The End-User Device(s) 109 uses a License Watermark 527 to embed the copy/play code within the Content 113. Only the End-User Player Application 195 that is knowledgeable of the embedding algorithm and the associated scrambling key is able (decryption 539) to read or modify the embedded data. The data is invisible or inaudible to a human observer; that is, the data introduces no perceivable degradation to the Content 113. Since the watermark survives several steps of content processing, data compression, D-to-A and A-to-D conversion, and signal degradation introduced by normal content handling, the watermark stays with the Content 113 in any representation form, including analog representation. In an alternate embodiment, instead of using a License Watermark 527 to embed the copy/play code within the Content 113, the End-User Player Application 195 uses securely stored Usage Conditions 519.

3. Content Identification Layer 503

As part of the Content Identification Layer 503, the Content Provider(s) 101 also uses a License Watermark 527 to embed data in the Content 113 such as to the content identifier, content owner and other information, such as publication date and geographic distribution region. This watermark is referred to here as the Copyright Watermark 529. Upon reception, the End-User Device(s) 109 watermarks the copy of the Content 113 with the content purchaser's name and the Transaction ID 535 (see the License Control Layer 501 section below), and with other information

such as date of license and Usage Conditions 517. This watermark is referred to here as the license watermark. Any copy of Content 113, obtained in an authorized manner or not, and subject to audio processing that preserves the content quality, carries the copyright and license watermarks. The Content Identification Layer 503 deters piracy.

4. License Control Layer 501

The License Control Layer 501 protects the Content 113 against unauthorized interception and ensures that the Content is only released on an individual basis to an End-User(s) that has properly licensed End-User Device(s) 109 and successfully completes a license purchase transaction with an authorized Electronic Digital Content Store(s) 103. The License Control Layer 501 protects the Content 113 by double Encryption 531. The Content 113 is encrypted using an encryption symmetric key generated by the Content Provider(s) 101, and the symmetric key is encrypted using the public key 621 of the Clearinghouse(s). Only the Clearinghouse(s) 105 can initially recover the symmetric key.

License control is designed with the Clearinghouse(s) 105 as the "trusted party". Before releasing permission for the License Request 537, (i.e. the Symmetric Key 623 for the Content 113 to an End-User Device(s) 109), the Clearinghouse(s) 105 verifies that the Transaction 541 and the License Authorization 543 are complete and authentic, that the Electronic Digital Content Store(s) 103 has authorization from the Secure Digital Content Electronic Distribution System 100 for the sale of electronic Content 113, and that the End-User(s) has a properly licensed application. Audit/Reporting 545 allows the generation of reports and the sharing of licensing transaction information with other authorized parties in the Secure Electronic Digital Content Distribution System 100

License control is implemented through SC Processing 533. SC(s) are used to distribute encrypted Content 113 and information among the system operation components (more about the SC(s) detailed structure sections below). A SC is cryptographic carrier of information that uses cryptographic encryption, digital signatures and digital certificates to provide protection against unauthorized interception and modification of the electronic information or Content 113. It also allows for the authenticity verification of the electronic data.

License control requires that the Content Provider(s) 101, the Electronic Digital Content Store(s) 103, and the Clearinghouse(s) 105 have bona-fide cryptographic digital certificates from reputable Certificate Authorities that are used to authenticate those components. The End-User Device(s) 109 are not required to have digital certificates.

5

C. Content Distribution and Licensing Control

FIG. 6 is a block diagram illustrating an overview of the Content Distribution and Licensing Control as it applies to the License Control Layer of FIG. 5. The figure depicts the case in which the Electronic Digital Content Store(s) 103, End-User Device(s) 109 and the Clearinghouse(s) 105 are
10 interconnected via the Internet, and unicast (point-to-point) transmission is used among those components. The communication between the Content Provider(s) 101 and the Electronic Digital Content Store(s) 103 could also be over the Internet or other network. It is assumed that the Content-purchase commercial transaction between the End-User Device(s) 109 and the Electronic Digital Content Store(s) 103 is based on standard Internet Web protocols. As part of the Web-based
15 interaction, the End-User(s) makes the selection of the Content 113 to purchase, provides personal and financial information, and agrees to the conditions of purchase. The Electronic Digital Content Store(s) 103 could obtain payment authorization from an acquirer institution using a protocol such as SET.

It is also assumed in FIG. 6 that the Electronic Digital Content Store(s) 103 has downloaded
20 the End-User Player Application 195 to an End-User Device(s) 109 based on standard Web protocols. The architecture requires that the Electronic Digital Content Store(s) 103 assigns a unique application ID to the downloaded Player Application 195 and that the End-User Device(s) 109 stores it for later application license verification (see below).

The overall licensing flow starts at the Content Provider(s) 101. The Content Provider(s) 101
25 encrypts the Content 113 using an encryption symmetric key locally generated, and encrypts the Symmetric Key 623 using the Clearinghouse's 105 public key 621. In an alternate embodiment, the symmetric key instead of being locally generated my be sent to the Content Provider(s) 101 from the Clearinghouse(s) 105. The Content Provider(s) 101 creates a Content SC(s) 630 around the encrypted Content 113, and a Metadata SC(s) 620 around the encrypted Symmetric Key 623, Store

Usage Conditions 519, and other Content 113 associated information. There is one Metadata SC(s) 620 and one Content SC(s) 630 for every Content 113 object. The Content 113 object may be a compression level one same song or the Content 113 object may be each song on the album or the Content 113 object may be the entire album. For each Content 113 object, the Metadata SC(s) 620 also carries the Store Usage Conditions 519 associated with the Content Usage Control Layer 505.

The Content Provider(s) 101 distributes the Metadata SC(s) 620 to one or more Electronic Digital Content Store(s) 103 (step 601) and the Content SC(s) 630 to one or more Content Hosting Sites (step 602). Each Electronic Digital Content Store(s) 103, in turn creates an Offer SC(s) 641. The Offer SC(s) 641 typically carries much of the same information as the Metadata SC(s) 620, including the Digital Signature 624 of the Content Provider(s) 101 and the Certificate (not shown of the Content Provider(s) 101. As mentioned above, the Electronic Digital Content Store(s) 103 can add to or narrow the Store Usage Conditions 519 (handled by the Control Usage Control Layer) initially defined by the Content Provider(s) 101. Optionally, the Content SC(s) 630 and/or the Metadata SC(s) 620 is signed with a Digital Signature 624 of the Content Provider(s) 101.

After the completion of the Content-purchase transaction between the End-User Device(s) 109 and the Electronic Digital Content Store(s) 103 (step 603), the Electronic Digital Content Store(s) 103 creates and transfers to the End-User Device(s) 109 a Transaction SC(s) 640 (step 604). The Transaction SC(s) 640 includes a unique Transaction ID 535, the purchaser's name (i.e. End-User(s)') (not shown), the Public Key 661 of the End-User Device(s) 109, and the Offer SC(s) 641 associated with the purchased Content 113. Transaction Data 642 in FIG. 6 represents both the Transaction ID 535 and the End-User(s) name (not shown). The Transaction Data 642 is encrypted with the Public Key 621 of the Clearinghouse(s) 105. Optionally, the Transaction SC(s) 640 is signed with a Digital Signature 643 of the Electronic Digital Content Store(s) 103.

Upon reception of the Transaction SC(s) 640 (and the Offer SC(s) 641 included in it), the End-User Player Application 195 running on End-User Device(s) 109 solicits license authorization from the Clearinghouse(s) 105 by means of an Order SC(s) 650 (step 605). The Order SC(s) 650 includes the encrypted Symmetric Key 623 and Store Usage Conditions 519 from the Offer SC(s) 641, the encrypted Transaction Data 642 from the Transaction SC(s) 640, and the encrypted Application ID 551 from the End-User Device(s) 109. In another embodiment, the Order SC(s) 650

is signed with a Digital Signature 652 of the End-User Device(s) 109.

Upon reception of the Order SC(s) 650 from the End-User Device(s) 109, the Clearinghouse(s) 105 verifies:

1. that the Electronic Digital Content Store(s) 103 has authorization from the Secure Digital Content Electronic Distribution System 100 (exists in the Database 160 of the Clearinghouse(s) 105);
2. that the Order SC(s) 650 has not been altered;
3. that the Transaction Data 642 and Symmetric Key 623 are complete and authentic;
4. that the electronic Store Usage Conditions 519 purchased by the End-User Device(s) 109 are consistent with those Usage Conditions 517 set by the Content Provider(s) 101; and
5. that the Application ID 551 has a valid structure and that it was provided by an authorized Electronic Digital Content Store(s) 103.

If the verifications are successful, the Clearinghouse(s) 105 decrypts the Symmetric Key 623 and the Transaction Data 642 and builds and transfers the License SC(s) 660 to the End-User Device(s) 109 (step 606).

The License SC(s) 660 carries the Symmetric Key 623 and the Transaction Data 642, both encrypted using the Public Key 661 of the End-User Device(s) 109. If any verification is not successful, the Clearinghouse(s) 105 denies the license to the End-User Device(s) 109 and informs the End-User Device(s) 109. The Clearinghouse(s) 105 also immediately informs (step 611) the Electronic Digital Content Store(s) 103 of this verification failure. In an alternate embodiment, the Clearinghouse(s) 105 signs the License SC(s) 660 with its Digital Signature 663.

After receiving the License SC(s) 660, the End-User Device(s) 109 decrypts the Symmetric Key 623 and the Transaction Data 642 previously received from the Clearinghouse(s) 105 and requests the Content SC(s) 630 (step 607) from a Content Hosting Site(s) 111. Upon arrival of the Content SC(s) 630 (step 608), the End-User Device(s) 109 decrypts the Content 113 using the Symmetric Key 623 (step 609), and passes the Content 113 and the Transaction Data 642 to the other layers for license watermarking, copy/play coding, scrambling, and further Content 113 processing as described previously for FIG. 5.

Finally, the Clearinghouse(s) 105 on a periodic basis transmits summary transaction reports

to the Content Provider(s) 101 and the Electronic Digital Content Store(s) 103 for auditing and tracking purposes (step 610).

V. SECURE CONTAINER STRUCTURE

A. General Structure

A Secure Container (SC) is a structure that consists of several parts which together define a unit of Content 113 or a portion of a transaction, and which also define related information such as Usage Conditions, metadata, and encryption methods. SC(s) are designed in such a way that the integrity, completeness, and authenticity of the information can be verified. Some of the information in SC(s) may be encrypted so that it can only be accessed after proper authorization has been obtained.

SC(s) include at least one bill of materials (BOM) part which has records of information about the SC(s) and about each of the parts included in the SC(s). A message digest is calculated, using a hashing algorithm such as MD-5, for each part and then included in the BOM record for the part. The digests of the parts are concatenated together and another digest is computed from them and then encrypted using the private key of the entity creating the SC(s) to create a digital signature. Parties receiving the SC(s) can use the digital signature to verify all of the digests and thus validate the integrity and completeness of the SC(s) and all of its parts.

The following information may be included as records in the BOM along with the records for each part. The SC(s) type determines which records need to be included:

- SC(s) version
- SC(s) ID
- Type of SC(s) (e.g. Offer, Order, Transaction, Content, Metadata or promotional and License.)
- Publisher of the SC(s)
- Date that the SC(s) was created
- Expiration date of the SC(s)
- Clearinghouse(s) URL
- Description of the digest algorithm used for the included parts (default is MD-5)

- Description of the algorithm used for the digital signature encryption (default is RSA)
- Digital signature (encrypted digest of all of the concatenated digests of the included parts)

SC(s) may include more than one BOM. For example, an Offer SC(s) 641 consists of the original Metadata SC(s) 620 parts, including its BOM, as well as additional information added by the Electronic Digital Content Store(s) 103 and a new BOM. A record for the Metadata SC(s) 620 BOM is included in the Offer SC(s) 641 BOM. This record includes a digest for the Metadata SC(s) 620 BOM which can be used to validate its integrity and therefore, the integrity of the parts included from the Metadata SC(s) 620 can also be validated using the part digest values stored in Metadata SC(s) 620 BOM. None of the parts from the Metadata SC(s) 620 have records in the new BOM that was created for the Offer SC(s) 641. Only parts added by the Electronic Digital Content Store(s) 103 and the Metadata SC(s) 620 BOM have records in the new BOM.

SC(s) may also include a Key Description part. Key Description parts include records that contain the following information about encrypted parts in the SC(s):

- The name of the encrypted part.
- The name to use for the part when it is decrypted.
- The encryption algorithm used to encrypt the part.
- Either a Key Identifier to indicate the public encryption key that was used to encrypt the part or an encrypted symmetric key that, when decrypted, is used to decrypt the encrypted part.
- The encryption algorithm used to encrypt the symmetric key. This field is only present when the record in the Key Description part includes an encrypted symmetric key that was used to encrypt the encrypted part.
- A Key Identifier of the public encryption key that was used to encrypt the symmetric key. This field is only present when the record in the Key Description part includes an encrypted symmetric key and the encryption algorithm identifier of the symmetric key that was used to encrypt the encrypted part.

If the SC(s) does not contain any encrypted parts, then there is no Key Description part.

B. Rights Management Language Syntax and Semantics

The Rights Management Language consists of parameters that can be assigned values to

define restrictions on the use of the Content 113 by an End-User(s) after the Content 113 purchase. The restrictions on the use of the Content 113 is the Usage Conditions 517. Each Content Provider(s) 101 specifies the Usage Conditions 517 for each of its Content 113 items. Electronic Digital Content Store(s) 103 interpret the Usage Conditions 517 in Metadata SC(s) 620 and use the information to provide select options they wish to offer their customers as well as add retail purchase information for the Content 113. After an End-User(s) has selected a Content 113 item for purchase, the End-User Device(s) 109 requests authorization for the Content 113 based on Store Usage Conditions 519. Before the Clearinghouse(s) 105 sends a License SC(s) 660 to the End-User(s), the Clearinghouse(s) 105 verifies that the Store Usage Conditions 519 being requested are in agreement with the allowable Usage Conditions 517 that were specified by the Content Provider(s) 101 in the Metadata SC(s) 620.

When an End-User Device(s) 109 receives the Content 113 that was purchased, the Store Usage Conditions 519 are encoded into that Content 113 using the Watermarking Tool or encoded in the securely stored Usage Conditions 519. The End-User Player Application 195 running on End-User Device(s) 109 insures that the Store Usage Conditions 519 that were encoded into the Content 113 are enforced.

The following are examples of Store Usage Conditions 519 for an embodiment where the Content 113 is music:

- Song is recordable.
- Song can be played n number of times.

C. Overview of Secure Container Flow and Processing

Metadata SC(s) 620 are built by Content Provider(s) 101 and are used to define Content 113 items such as songs. The Content 113 itself is not included in these SC(s) because the size of the Content 113 is typically too large for Electronic Digital Content Store(s) 103 and End-User(s) to efficiently download the containers just for the purpose of accessing the descriptive metadata. Instead, the SC(s) includes an external URL (Uniform Resource Locators) to point to the Content 113. The SC(s) also includes metadata that provides descriptive information about the Content 113 and any other associated data, such as for music, the CD cover art and/or digital audio clips in the

case of song Content 113.

Electronic Digital Content Store(s) 103 download the Metadata SC(s) 620, for which they are authorized, and build Offer SC(s) 641. In short, an Offer SC(s) 641 consists of some of the parts and the BOM from the Metadata SC(s) 620 along with additional information included by the Electronic Digital Content Store(s) 103. A new BOM for the Offer SC(s) 641 is created when the Offer SC(s) 641 is built. Electronic Digital Content Store(s) 103 also use the Metadata SC(s) 620 by extracting metadata information from them to build HTML pages on their web sites that present descriptions of Content 113 to End-User(s), usually so they can purchase the Content 113.

The information in the Offer SC(s) 641 that is added by the Electronic Digital Content Store(s) 103 is typically to narrow the selection of Usage Conditions 517 that are specified in the Metadata SC(s) 620 and promotional data such as a graphic image file of the store's logo and a URL to the store's web site. An Offer SC(s) 641 template in the Metadata SC(s) 620 indicates which information can be overridden by the Electronic Digital Content Store(s) 103 in the Offer SC(s) 641 and what, if any, additional information is required by the Electronic Digital Content Store(s) 103 and what parts are retained in the embedded Metadata SC(s) 620.

Offer SC(s) 641 are included in a Transaction SC(s) 640 when an End-User(s) decides to purchase Content 113 from an Electronic Digital Content Store(s) 103. The Electronic Digital Content Store(s) 103 builds a Transaction SC(s) 640 and includes Offer SC(s) 641 for each Content 113 item being purchased and transmits it to the End-User Device(s) 109. The End-User Device(s) 109 receives the Transaction SC(s) 640 and validates the integrity of the Transaction SC(s) 640 and the included Offer SC(s) 641.

An Order SC(s) 650 is built by the End-User Device(s) 109 for each Content 113 item being purchased. Information is included from the Offer SC(s) 641, from the Transaction SC(s) 640, and from the configuration files of the End-User Device(s) 109. Order SC(s) 650 are sent to the Clearinghouse(s) 105 one at a time. The Clearinghouse(s) 105 URL where the Order SC(s) 650 is included as one of the records in the BOM for the Metadata SC(s) 620 and included again in the Offer SC(s) 641.

The Clearinghouse(s) 105 validates and processes Order SC(s) 650 to provide the End-User Device(s) 109 with everything that is required to a License Watermark 527 and access purchased

Content 113. One of the functions of the Clearinghouse(s) 105 is to decrypt the Symmetric Keys 623 that are needed to decrypt the watermarking instructions from the Offer SC(s) 641 and the Content 113 from the Content SC(s) 630. An encrypted Symmetric Key 623 record actually contains more than the actual encrypted Symmetric Key 623. Before executing the encryption, the Content Provider(s) 101 may optionally append its name to the actual Symmetric Key 623. Having the Content Provider(s)' 101 name encrypted together with the Symmetric Key 623 provides security against a pirate Content Provider(s) 101 that has built its own Metadata SC(s) 620 and Content SC(s) 630 from legal SC(s). The Clearinghouse(s) 105 verifies that the name of the Content Provider(s) 101 encrypted together with the Symmetric Keys 623 matches the name of the Content Provider(s) 101 in the SC(s) certificate.

If there are any changes required to be made to the watermarking instructions by the Clearinghouse(s) 105, then the Clearinghouse(s) 105 decrypts the Symmetric Key 623 and then modifies the watermarking instructions and encrypts them again using a new Symmetric Key 623. The Symmetric Key 623 is then re-encrypted using the Public Key 661 of the End-User Device(s) 109. The Clearinghouse(s) 105 also decrypts the other Symmetric Keys 623 in the SC(s) and encrypts them again with the Public Key 661 of the End-User Device(s) 109. The Clearinghouse(s) 105 builds a License SC(s) 660 that includes the newly encrypted Symmetric Keys 623 and updated watermarking instructions and sends it to the End-User Device(s) 109 in response to the Order SC(s) 650. If the processing of the Order SC(s) 650 does not complete successfully, then the Clearinghouse(s) 105 returns to the End-User Device(s) 109 an HTML page or equivalent reporting the failure of the authorization process.

A License SC(s) 660 provides an End-User Device(s) 109 with everything that is needed to access a Content 113 item. The End-User Device(s) 109 requests the appropriate Content SC(s) 630 from the Content Hosting Site(s) 111. Content SC(s) 630 are built by Content Provider(s) 101 and include encrypted Content 113 and metadata parts. The End-User Player Application 195 uses the Symmetric Keys 623 from the License SC(s) 660 to decrypt the Content 113, metadata, and watermarking instructions. The watermarking instructions are then affixed into the Content 113 and the Content 113 is scrambled and stored on the End-User Device(s) 109.

D. Metadata Secure Container 620 Format

The following table shows the parts that are included in a Metadata SC(s) 620. Each box in the Parts column is a separate object included in the SC(s) along with the BOM (with the exception of part names that are surrounded by [] characters). The BOM contains a record for each part included in the SC(s). The Part Exists column indicates whether the part itself is actually included in the SC(s) and the Digest column indicates whether a message digest is computed for the part. Some parts may not be propagated when a SC(s) is included in other SC(s) (as determined by the associated template), although the entire original BOM is propagated. This is done because the entire BOM is required by the Clearinghouse(s) 105 to verify the digital signature in the original SC(s).

The Key Description Part columns of the following table define the records that are included in the Key Description part of the SC(s). Records in the Key Description part define information about the encryption keys and algorithms that were used to encrypt parts within the SC(s) or parts within another SC(s). Each record includes the encrypted part name and, if necessary, a URL that points to another SC(s) that includes the encrypted part. The Result Name column defines the name that is assigned to the part after it is decrypted. The Encrypt Alg column defines the encryption algorithm that was used to encrypt the part. The Key Id/Enc Key column defines either an identification of the encryption key that was used to encrypt the part or a base64 encoding of the encrypted Symmetric Key 623 bit string that was used to encrypt the part. The Sym Key Alg column is an optional parameter that defines the encryption algorithm that was used to encrypt the Symmetric Key 623 when the previous column is an encrypted Symmetric Key 623. The Sym Key ID column is an identification of the encryption key that was used to encrypt the Symmetric Key 623

when the Key Id/Enc Key column is an encrypted Symmetric Key 623.

Parts	*BOM*		*Key Description Part*				
	Part Exists	*Digest*	*Result Name*	*Encrypt Alg*	*Key Id/Enc Key*	*Sym Key Alg*	*Sym Key ID*
[Content URL]			Output Part	RC4	Enc Sym Key	RSA	CH Pub Key
[Metadata URL]			Output Part	RC4	Enc Sym Key	RSA	CH Pub Key
	SC Version						
	SC ID						
	SC Type						
	SC Publisher						
	Date						
	Expiration Date						
	Clearinghouse(s) URL						
	Digest Algorithm ID						
	Digital Signature Alg ID						
Content ID	Yes	Yes					
Metadata	Yes	Yes					
Usage Conditions	Yes	Yes					
SC Templates	Yes	Yes					
Watermarking Instructions	Yes	Yes	Output Part	RC4	Enc Sym Key	RSA	CH Pub Key
Key Description Part	Yes	Yes					
Clearinghouse(s) Certificate(s)	Yes	No					
Certificate(s)	Yes	No					
	Digital Signature						

The following describes the terms that are used in the above Metadata SC(s) table:

- [Content URL] - A parameter in a record in the Key Description part. This is a URL that points to the encrypted Content 113 in the Content SC(s) 630 that is associated with this Metadata SC(s) 620. The Metadata SC(s) 620 itself does not contain the encrypted Content 113.
- [Metadata URL] - A parameter in a record in the Key Description part. This is a URL that points to the encrypted metadata in the Content SC(s) 630 that is associated with this Metadata SC(s) 620. The Metadata SC(s) 620 itself does not contain the encrypted metadata.
- Content ID - A part that defines a unique ID assigned to a Content 113 item. There is more than one Content ID included in this part if the Metadata SC(s) 620 references more than one Content 113 item.
- Metadata - Parts that contain information related to a Content 113 item such as the artist name and CD cover art in the case of a song. There may be multiple metadata parts, some of which may be encrypted. The internal structure of the metadata parts is dependent on the type of metadata contained therein.
- Usage Conditions - A part that contains information that describes usage options, rules, and

restrictions to be imposed on an End-User(s) for use of the Content 113.

- SC(s) Templates - Parts that define templates that describe the required and optional information for building the Offer, Order, and License SC(s) 660.
- Watermarking Instructions - A part that contains the encrypted instructions and parameters for implementing watermarking in the Content 113. The watermarking instructions may be modified by the Clearinghouse(s) 105 and returned back to the End-User Device(s) 109 within the License SC(s) 660. There is a record in the Key Description part that defines the encryption algorithm that was used to encrypt the watermarking instructions, the output part name to use when the watermarking instructions are decrypted, a base64 encoding of the encrypted Symmetric Key 623 bit string that is was used to encrypt the watermarking instructions, the encryption algorithm that was used to encrypt the Symmetric Key 623, and the identification of the public key that is required to decrypt the Symmetric Key 623.
- Clearinghouse(s) Certificate(s) - A certificate from a certification authority or from the Clearinghouse(s) 105 that contains the signed Public Key 621 of the Clearinghouse(s) 105. There may be more than one certificate, in which case a hierarchical level structure is used with the highest level certificate containing the public key to open the next lowest level certificate is reached which contains the Public Key 621 of the Clearinghouse(s) 105.
- Certificate(s) - A certificate from a certification authority or from the Clearinghouse(s) 105 that contains the signed Public Key 621 of the entity that created the SC(s). There may be more than one certificate, in which case a hierarchical level structure is used with the highest level certificate containing the public key to open the next level certificate, and so on, until the lowest level certificate is reached which contains the public key of the SC(s) creator.
- SC Version - A version number assigned to the SC(s) by the SC Packer Tool.
- SC ID - A unique ID assigned to the SC(s) by the entity that created the SC(s).
- SC Type - Indicates the type of SC(s) (e.g. Metadata, Offer, Order, etc.)
- SC Publisher - Indicates the entity that created the SC(s).
- Creation Date - Date that the SC(s) was created.
- Expiration Date - Date the SC(s) expires and is no longer valid.
- Clearinghouse(s) URL - Address of the Clearinghouse(s) 105 that the End-User Player

Application 195 should interact with to obtain the proper authorization to access the Content 113.

- Digest Algorithm ID - An identifier of the algorithm used to compute the digests of the parts.
- Digital Signature Alg ID - An identifier of the algorithm used to encrypt the digest of the concatenated part digests. This encrypted value is the digital signature.
- Digital Signature - A digest of the concatenated part digests encrypted with the public key of the entity that created the SC(s).
- Output Part - The name to assign to the output part when an encrypted part is decrypted.
- RSA and RC4 - Default encryption algorithms used to encrypt the Symmetric Keys 623 and data parts.
- Enc Sym Key - A base64 encoding of an encrypted key bitstring that, when decrypted, is used to decrypt a SC(s) part.
- CH Pub Key - An identifier that indicates that the Clearinghouse's 105 Public Key 621 was used to encrypt the data.

E. Offer Secure Container 641 Format

The following table shows the parts that are included in the Offer SC(s) 641. The parts, with the exception of some of the metadata parts, and BOM from the Metadata SC(s) 620 are also

included in the Offer SC(s) 641.

Parts	*BOM*			*Key Description Part*				
	Part Exists		*Digest*	*Result Name*	*Encrypt Alg*	*Key ID/Enc Key*	*Sym Key Alg*	*Sym Key ID*
---------- Metadata SC Parts ----------								
[Content URL]				Output Part	RC4	Enc Sym Key	RSA	CH Pub Key
[Metadata URL]				Output Part	RC4	Enc Sym Key	RSA	CH Pub Key
	SC Version							
	SC ID							
	SC Type							
	SC Publisher							
	Date							
	Expiration Date							
	Clearinghouse(s) URL							
	Digest Algorithm ID							
	Digital Signature Alg ID							
Content ID	Yes	Yes						
Metadata	Some	Yes						
Usage Conditions	Yes	Yes						
SC Templates	Yes	Yes						
Watermarking Instructions	Yes	Yes		Output Part	RC4	Enc Sym Key	RSA	CH Pub Key
Key Description Part	Yes	Yes						
Clearinghouse(s) Certificate(s)	Yes	No						
Certificate(s)	Yes	No						
	Digital Signature							
---------- Offer SC Parts ----------								
	SC Version							
	SC ID							
	SC Type							
	SC Publisher							
	Date							
	Expiration Date							
	Digest Algorithm ID							
	Digital Signature Alg ID							
Metadata SC BOM	Yes	Yes						
Additional and Overridden Fields	Yes	Yes						
Electronic Digital Content Store(s) Certificate	Yes	No						
Certificate(s)	Yes	No						
	Digital Signature							

The following describes the terms that are used in the above Offer SC(s) 641 that were not previously described for another SC(s):

- Metadata SC(s) BOM - The BOM from the original Metadata SC(s) 620. The record in the Offer SC(s) 641 BOM includes the digest of the Metadata SC(s) 620 BOM.
- Additional and Overridden Fields - Usage conditions information that was overridden by the Electronic Digital Content Store(s) 103. This information is validated by the Clearinghouse(s) 105, by means of the received SC(s) templates, to make sure that anything that the Electronic Digital Content Store(s) 103 overrides is within the scope of its authorization.

- Electronic Digital Content Store(s) Certificate - A certificate provided to the Electronic Digital Content Store(s) 103 by the Clearinghouse(s) 105 and signed by the Clearinghouse(s) 105 using its private key. This certificate is used by the End-User Player Application 195 to verify that the Electronic Digital Content Store(s) 103 is a valid distributor of Content 113. The End-User Player Application 195 and Clearinghouse(s) 105 can verify that the Electronic Digital Content Store(s) 103 is an authorized distributor by decrypting the certificate's signature with the Clearinghouse's 105 Public Key 621. The End-User Player Application 195 keeps a local copy of the Clearinghouse's 105 Public Key 621 that it receives as part of its initialization during installation.

F. Transaction Secure Container 640 Format

The following table shows the parts that are included in the Transaction SC(s) 640 as well as its BOM and Key Description parts.

Parts	*BOM*		*Key Description Part*				
	Part Exists	*Digest*	*Result Name*	*Encrypt Alg*	*Key ID/Enc Key*	*Sym Key Alg*	*Sym Key ID*
	SC Version						
	SC ID						
	SC Type						
	SC Publisher						
	Date						
	Expiration Date						
	Digest Algorithm ID						
	Digital Signature Alg ID						
Transaction ID	Yes	Yes	Output Part	RSA	CH Pub Key		
End-User(s) ID	Yes	Yes	Output Part	RSA	CH Pub Key		
End-User(s)' Public Key	Yes	Yes					
Offer SC(s)	Yes	Yes					
Selections of Content Use	Yes	Yes					
HTML to Display	Yes	Yes					
Key Description Part	Yes	Yes					
Electronic Digital Content Store(s) Certificate	Yes	No					
	Digital Signature						

The following describes the terms that are used in the above Transaction SC(s) 640 that were not previously described for another SC(s):

- Transaction ID 535 - An ID assigned by the Electronic Digital Content Store(s) 103 to uniquely identify the transaction.
- End-User(s) ID - An identification of the End-User(s) obtained by the Electronic Digital

Content Store(s) 103 at the time the End-User(s) makes the buying selection and provides the credit card information.

- End-User(s)' Public Key - The End-User(s)' Public Key 661 that is used by the Clearinghouse(s) 105 to re-encrypt the Symmetric Keys 623. The End-User(s)' Public Key 661 is transmitted to the Electronic Digital Content Store(s) 103 during the purchase transaction.
- Offer SC(s) - Offer SC(s) 641 for the Content 113 items that were purchased.
- Selections of Content Use - An array of Usage Conditions for each Content 113 item being purchased by the End-User(s). There is an entry for each Offer SC(s) 641.
- HTML to Display - One or more HTML pages that the End-User Player Application 195 displays in the Internet browser window upon receipt of the Transaction SC(s) 640 or during the interaction between the End-User Device(s) 109 and the Clearinghouse(s) 105.

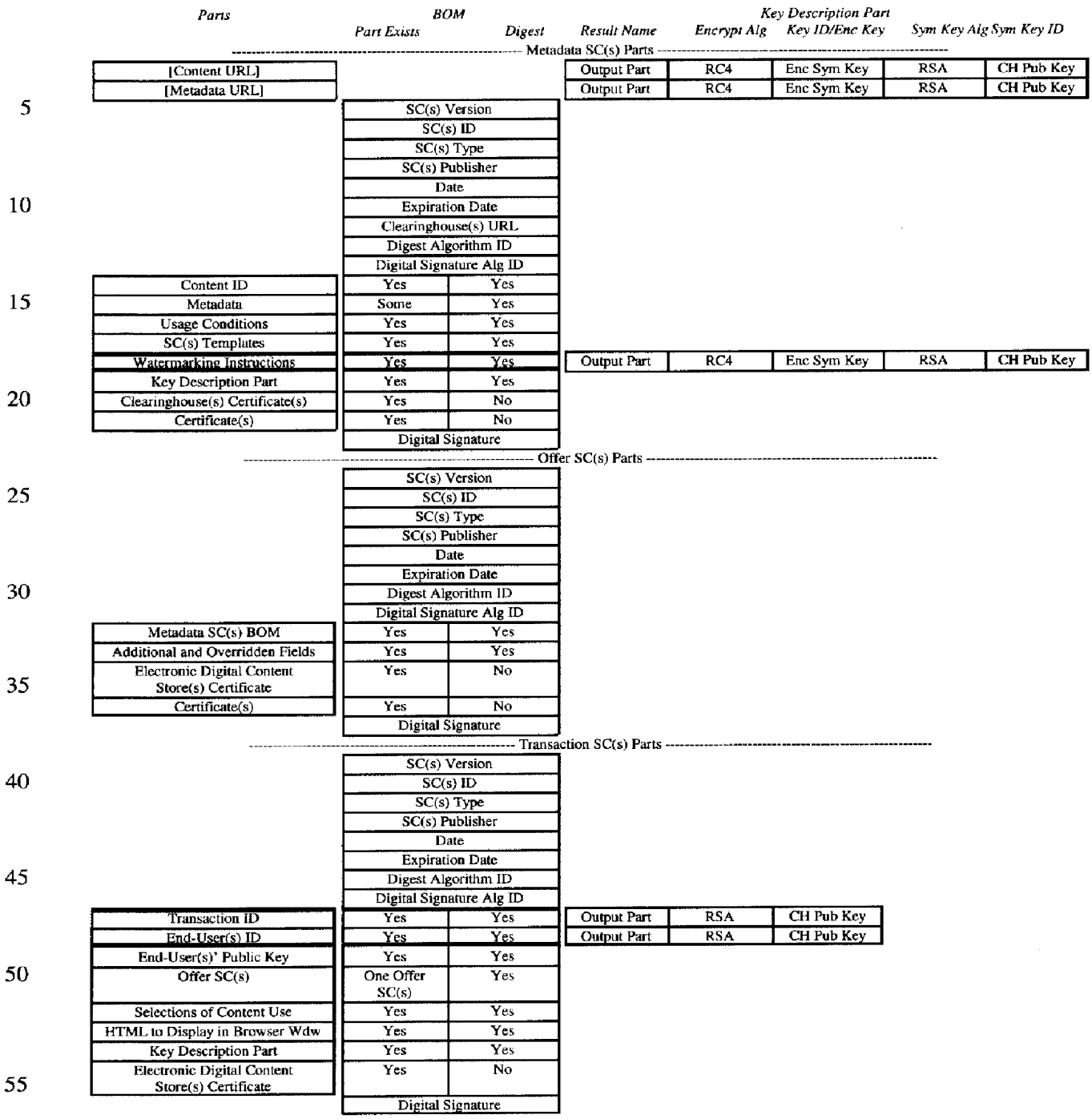
When the End-User Device(s) 109 receives a Transaction SC(s) 640, the following steps may be performed to verify the integrity and authenticity of the SC(s):

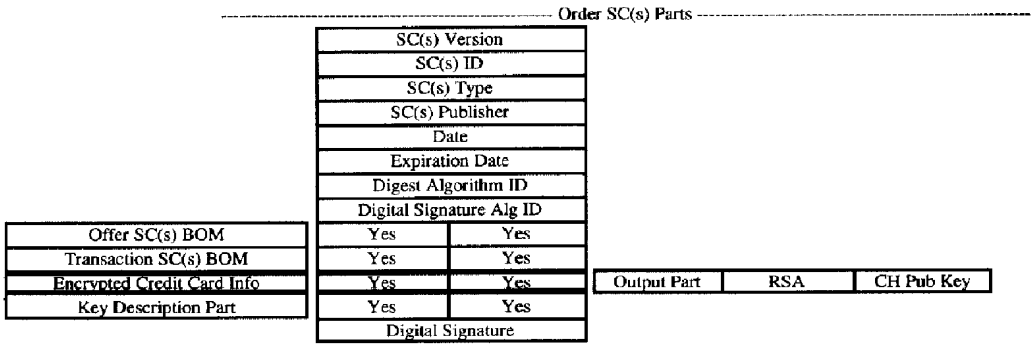
1. Verify the integrity of the Electronic Digital Content Store(s) 103 certificate using the Public Key 621 of the Clearinghouse(s) 105. The Public Key 621 of the Clearinghouse(s) 105 was stored at the End-User Device(s) 109 after it was received as part of the initialization of the End-User Player Application 195 during its installation process.
2. Verify the Digital Signature 643 of the SC(s) using the public key from the Electronic Digital Content Store(s) 103 certificate.
3. Verify the hashes of the SC(s) parts.
4. Verify the integrity and authenticity of each Offer SC(s) 641 included in the Transaction SC(s) 640.

G. Order Secure Container 650 Format

The following table shows the parts that are included in the Order SC(s) 650 as well as its BOM and Key Description parts. These parts either provide information to the Clearinghouse(s) 105 for decryption and verification purposes or is validated by the Clearinghouse(s) 105. The parts and

BOM from the Offer SC(s) 641 are also included in the Order SC(s) 650. The Some string in the Part Exists column of the Metadata SC(s) BOM indicates that the some of those parts are not included in the Order SC(s) 650. The BOM from the Metadata SC(s) 620 is also included without any change so that the Clearinghouse(s) 105 can validate the integrity of the Metadata SC(s) 620 and its parts.


Parts
BOM
Part Exists
Digest
Result Name
Key Description Part
Encrypt Alg
Key ID/Enc Key
Sym Key Alg
Sym Key ID
Metadata SC(s) Parts
[Content URL]
[Metadata URL]
Output Part
RC4
Enc Sym Key
RSA
CH Pub Key
SC(s) Version
SC(s) ID
SC(s) Type
SC(s) Publisher
Date
Expiration Date
Clearinghouse(s) URL
Digest Algorithm ID
Digital Signature Alg ID
Content ID
Metadata
Usage Conditions
SC(s) Templates
Watermarking Instructions
Key Description Part
Clearinghouse(s) Certificate(s)
Certificate(s)
Digital Signature
Offer SC(s) Parts
Metadata SC(s) BOM
Additional and Overridden Fields
Electronic Digital Content Store(s) Certificate
Transaction SC(s) Parts
Transaction ID
End-User(s) ID
End-User(s)' Public Key
Offer SC(s)
Selections of Content Use
HTML to Display in Browser Wdw



The following describes the terms that are used in the above Order SC(s) 650 that were not previously described for another SC(s):

- Transaction SC(s) BOM - The BOM in the original Transaction SC(s) 640. The record in the Order SC(s) 650 BOM includes the digest of the Transaction SC(s) 640 BOM.
- Encrypted Credit Card Info.- Optional encrypted information from the End-User(s) that is used to charge the purchase to a credit card or debit card. This information is required when the Electronic Digital Content Store(s) 103 that created the Offer SC(s) 641 does not handle the customer billing, in which case the Clearinghouse(s) 105 may handle the billing.

H. License Secure Container 660 Format

The following table shows the parts that are included in the License SC(s) 660 as well as its BOM. As shown in the Key Description part, the Symmetric Keys 623 that are required for decrypting the watermarking instructions, Content 113, and Content 113 metadata have been re-encrypted by the Clearinghouse(s) 105 using the End-User(s)' Public Key 661. When the End-User Device(s) 109 receives the License SC(s) 660 it decrypts the Symmetric Keys 623 and use them to access the encrypted parts from the License SC(s) 660 and the Content SC(s) 630.

Parts	*BOM*		*Key Description Part*				
	Part Exists	*Digest*	*Result Name*	*Encrypt Alg*	*Key ID/Enc Key*	*Sym Key Alg*	*Sym Key ID*
[Content URL]			Output Part	RC4	Enc Sym Key	RSA	EU Pub Key
[Metadata URL]			Output Part	RC4	Enc Sym Key	RSA	EU Pub Key
	SC(s) Version						
	SC(s) ID						
	SC(s) Type						
	SC(s) Publisher						
	Date						
	Expiration Date						
	Digest Algorithm ID						
	Digital Signature Alg ID						
Content ID	Yes	Yes					
Usage Conditions	Yes	Yes					
Transaction Data	Yes	Yes					
Watermarking Instructions	Yes	Yes	Output Part	RC4	Enc Sym Key	RSA	EU Pub Key
Key Description Part	Yes	Yes					
Certificate(s)	Yes	No					
	Digital Signature						

The following describes the terms that are used in the above License SC(s) 660 that were not previously described for another SC(s):

- EU Pub Key - An identifier that indicates that the End-User(s)' Public Key 661 was used to encrypt the data.
- Order SC(s) 650 ID - The SC(s) ID taken from the Order SC(s) 650 BOM.
- Certificate Revocation List – An optional list of certificate IDs which were previously issued and signed by the Clearinghouse(s) 105, but are no longer considered to be valid. Any SC(s) that have a signature which can be verified by a certificate that is included in the revocation list are invalid SC(s). The End-User Player Application 195 stores a copy of the Clearinghouse's 105 certificate revocation list on the End-User Device(s) 109. Whenever a revocation list is received, the End-User Player Application 195 replaces its local copy if the new one is more up to date. Revocation lists includes a version number or a time stamp (or both) in order to determine which list is the most recent.

I. Content Secure Container Format

The following table shows the parts that are included in the Content SC(s) 630 as well as the BOM:

Parts	BOM	
	Part Exists	Digest
	SC(s) Version	
	SC(s) ID	
	SC(s) Type	
	SC(s) Publisher	
	Date	
	Expiration Date	
	Clearinghouse(s) 105 URL	
	Digest Algorithm ID	
	Digital Signature Alg ID	
Content ID	Yes	Yes
Encrypted Content	Yes	Yes
Encrypted Metadata	Yes	Yes
Metadata	Yes	Yes
Certificate(s)	Yes	No
	Digital Signature	

The following describes the terms used in the above Content SC(s) 630 that were not previously described for another SC(s):

- Encrypted Content - Content 113 that was encrypted by a Content Provider(s) 101 using a Symmetric Key 623.
- Encrypted Metadata - Metadata associated with the Content 113 that was encrypted by a Content Provider(s) 101 using a Symmetric Key 623.

There is no Key Description part included in the Content SC(s) 630 since the keys required to decrypt the encrypted parts are in the License SC(s) 660 that is built at the Clearinghouse(s) 105.

VI. SECURE CONTAINER PACKING AND UNPACKING

A. Overview

The SC(s) Packer is a 32-bit Windows' program with an API (Application Programming Interface) that can be called in either a multiple or single step process to create a SC(s) with all of the specified parts. The SC(s) Packer 151, 152, 153 variety of hardware platforms supporting Windows' program at the Content Provider(s) 101, Clearinghouse(s) 105, Electronic Digital Content Store(s) 103 and other sites requiring SC(s) Packing. A BOM and, if necessary, a Key Description part are created and included in the SC(s). A set of packer APIs allows the caller to specify the information required to generate the records in the BOM and Key Description parts and to include parts in the SC(s). Encryption of parts and Symmetric Keys 623 as well as computing the digests

and the digital signature is also be performed by the packer. Encryption and digest algorithms that are supported by the packer are included in the packer code or they are called through an external interface.

The interface to the packer for building a SC(s) is done by an API that accepts the following parameters as input:

- A pointer to a buffer of concatenated structures. Each structure in the buffer is a command to the packer with the information that is required to execute the command. Packer commands include adding a part to the SC(s) with an associated BOM record, adding a record to the BOM, and adding records to the Key Description part.
- A value indicating the number of concatenated structures contained in the above described buffer.
- Name and location of the BOM part.
- A value with each bit being a defined flag or a reserved flag for future use. The following flags are currently defined:
  - Indication as to whether all of the parts of the SC(s) should be bundled together into a single file after all of the structures in the buffer have been processed. Bundling the parts into a single object is the last step that is performed when building a SC(s).
  - Indication as to whether the digital signature is omitted from the BOM part. If this flag is not set, then the digital signature is computed right before the SC(s) is bundled into a single object.

In an alternate embodiment, the interface to the packer for building a SC(s) is done by APIs that accept the following parameters as input:

- First, an API is called to create a Bill of Materials (BOM) part by passing in pointer to a structure that consists of information that is used to initialize SC(s) settings that are denoted as IP records in the SC(s) BOM part, the name to use for the BOM part, a default location to look for parts that will be added, and a flags value. This API returns a SC(s) handle that is used in subsequent Packer APIs.
- The Packer has an API that is used whenever a part is added to a SC(s). This API accepts a SC(s) handle, which was previously returned by a previous Packer API, a pointer to a

structure that consists of information about the part that is being added, and a flags value. Information about the part being added includes the name and location of the part, the name to use in the BOM for the part, the type of part that is being added, a hash value for the part, flags, etc.

- After all of the parts have been added to the SC(s) a Packer API is called to pack all of the parts, including the BOM part, into a single SC(s) object, which is typically a file. This API accepts a SC(s) handle, which was previously returned by a previous Packer API, the name to use for the packed SC(s), a pointer to a structure with information for signing the SC(s), and a flags value.

Either the packer or the entity calling the packer can use a SC(s) template to build a SC(s). SC(s) templates have information that define parts and records that are required in the SC(s) that is being built. Templates can also define encryption methods and key references to use for encrypting Symmetric Keys 623 and encrypted parts.

The packer has an API that is used to unpack a SC(s). Unpacking a SC(s) is the process of taking a SC(s) and separating it into its individual parts. The packer can then be called to decrypt any of the encrypted parts that were unpacked from the SC(s).

B. Bill of Materials (BOM) Part

The BOM part is created by the packer when a SC(s) is being built. The BOM is a text file that contains records of information about the SC(s) and about the parts that are included in the SC(s). Each record in the BOM is on a single line with a new line indicating the start of a new record. The BOM usually includes digests for each part and a digital signature that can be used to validate the authenticity and integrity of the SC(s).

The record types within a BOM are as follows:

IP An IP record contains a set of Name=Value pairs pertaining to the SC(s). The following Names are reserved for specific properties of SC(s):

V major.minor.fix

The V property specifies the version of the SC(s). This is the version number of the SC(s) specification that the SC(s) was created under. The string that follows should be of the form major.minor.fix, where major, minor, and fix are the major release number, minor release number, and fix level, respectively.

ID value

The ID property is a unique value that is assigned to this specific SC(s) by the entity that is creating this SC(s). The format of the value is defined in a later version of this document.

T value

The T property specifies the type of the SC(s), which should be one of:

ORD - An Order SC(s) 650.

OFF - An Offer SC(s) 641.

LIC - A License SC(s).

TRA - A Transaction SC(s) 640.

MET - A Metadata SC(s) 620.

CON - A Content SC(s) 630.

A value

The A property identifies the author or publisher of the SC(s). Author/publisher identities should be unambiguous and/or registered with the Clearinghouse(s) 105.

D value

The D property identifies the date, and optionally, the time that the SC(s) was created. The value should be of the form yyyy/mm/dd[@hh:mm[:ss[.fsec]][(TZ)]] representing year/month/day@hour:minute:second.decimal-fraction-of-second (time-zone). Optional parts of the value are enclosed in [] characters.

E value

The E property identifies the date, and optionally, the time that the SC(s) expires. The value should be the same form used in the D property that was previously defined. The expiration date/time should be compared, whenever possible, with the date/time at the Clearinghouse(s) 105.

CCURL value
The CCURL property identifies the URL of the Clearinghouse(s) 105. The value should be of the form of a valid external URL.

H value
The H property identifies the algorithm that was used to calculate the message digests for the parts included in the SC(s). An example digest algorithm is MD5.

D A D record is a data or part entry record that contains information that identifies the type of part, the name of the part, the (optional) digest of the part, and an (optional) indication that the part is not included in the SC(s). A - sign immediately after the type identifier is used to indicate that the part is not included in the SC(s). The following are reserved types of data or part records:

K part_name [digest]
Specifies the Key Description part.

W part_name [digest]
Specifies the watermarking instructions part.

C part_name [digest]
Specifies the certificate(s) used to validate the digital signature.

T part_name [digest]

Specifies the Usage Conditions part.

YF part_name [digest]

Specifies the Template part for the Offer SC(s) 641.

YO part_name [digest]

Specifies the Template part for the Order SC(s) 650.

YL part_name [digest]

Specifies the Template part for the License SC(s) 660.

ID part_name [digest]

Specifies the ID(s) of the Content 113 of the item(s) of Content 113 being referenced.

CH part_name [digest]

Specifies the Clearinghouse(s) 105 certificate part.

SP part_name [digest]

Specifies the Electronic Digital Content Store(s) 103 certificate part.

B part_name [digest]

Specifies a BOM part for another SC(s) that has its parts or a subset of its parts included in this SC(s).

BP part_name sc_part_name [digest]

Specifies a BOM part for another SC(s) that is included as a single part in this SC(s). The sc_part_name parameter is the name of the SC(s) part that is included in this SC(s) and that this BOM part defines. A BOM that is identical to this one is also included in the SC(s) that is defined by the sc_part_name parameter.

D part_name [digest]

Specifies a data (or metadata) part.

S An S record is a signature record the is used to define the digital signature of the SC(s). The digital signature is specified as follows:

S key_identifier signature_string signature_algorithm

The S record contains the key_identifier to indicate the encryption key of the signature, the signature_string, which is the base64 encoding of the digital signature bitstring, and the signature algorithm that was used to encrypt the digest to create the digital signature.

C. Key Description Part

The Key Description part is created by the packer to provide information about encryption keys that are needed for decryption of SC(s) encrypted parts. The encrypted parts may be included in the SC(s) being built or may be in other SC(s) which are referred to by the SC(s) being built. The Key Description part is a text file that contains records of information about the encryption keys and the parts for which the encryption keys are used. Each record in the Key Description part is on a single line with a new line indicating the start of a new record.

The following record type is used within a Key Description part and is defined as follows:

K encrypted_part_name; result_part_name; part_encryption_algorithm_identifier; public_key_identifier key_encryption_algorithm and encrypted_symmetric_key.

A K record specifies an encrypted part that may be included in this SC(s) or may be included in another SC(s) that is referred to by this record. The encrypted_part_name is either the name of a part in this SC(s) or a URL pointing to the name of the encrypted part in another SC(s). The result_part_name is the name that is given to the decrypted part. The part_encryption_algorithm_identifier indicates the encryption algorithm that was used to

encrypt the part. The public_key_identifier is an identifier of the key that was used to encrypt the Symmetric Key 623.

The key_encryption_algorithm_identifier indicates the encryption algorithm that was used to encrypt the Symmetric Key 623. The encrypted symmetric key is a base64 encoding of the encrypted Symmetric Key 623 bit string that was used to encrypt the part.

VII. CLEARINGHOUSE(S) 105

A. Overview

The Clearinghouse(s) 105 is responsible for the rights management functions of the Secure Digital Content Electronic Distribution System 100. Clearinghouse(s) 105 functions include enablement of Electronic Digital Content Store(s) 103, verification of rights to Content 113, integrity and authenticity validation of the buying transaction and related information, distribution of Content encryption keys or Symmetric Keys 623 to End-User Device(s) 109, tracking the distribution of those keys, and reporting of transaction summaries to Electronic Digital Content Store(s) 103 and Content Provider(s) 101. Content encryption keys are used by End-User Device(s) 109 to unlock Content 113 for which they have obtained rights, typically by a purchase transaction from an authorized Electronic Digital Content Store(s) 103. Before a Content encryption key is sent to an End-User Device(s) 109, the Clearinghouse(s) 105 goes through a verification process to validate the authenticity of the entity that is selling the Content 113 and the rights that the End-User Device(s) 109 has to the Content 113. This is called the SC Analysis Tool 185. In some configurations the Clearinghouse(s) 105 may also handle the financial settlement of Content 113 purchases by co-locating a system at the Clearinghouse(s) 105 that performs the Electronic Digital Content Store(s) 103 functions of credit card authorization and billing. The Clearinghouse(s) 105 uses OEM packages such as ICVerify and Taxware to handle the credit card processing and local sales taxes.

Electronic Digital Content Store(s) Embodiment

An Electronic Digital Content Store(s) 103 that wants to participate as a seller of Content 113 in the Secure Digital Content Electronic Distribution System 100 makes a request to one or more of

the Digital Content Provider(s) 101 that provide Content 113 to the Secure Digital Content Electronic Distribution System 100. There is no definitive process for making the request so long as the two parties come to an agreement. After the digital content label such as a Music Label e.g. Sony, Time-Warner, etc. decides to allow the Electronic Digital Content Store(s) 103 to sell its Content 113, the Clearinghouse(s) 105 is contacted, usually via E-mail, with a request that the Electronic Digital Content Store(s) 103 be added to the Secure Digital Content Electronic Distribution System 100. The digital content label provides the name of the Electronic Digital Content Store(s) 103 and any other information that may be required for the Clearinghouse(s) 105 to create a digital certificate for the Electronic Digital Content Store(s) 103. The digital certificate is sent to the digital content label in a secure fashion, and then forwarded by the digital content label to the Electronic Digital Content Store(s) 103. The Clearinghouse(s) 105 maintains a database of digital certificates that it has assigned. Each certificate includes a version number, a unique serial number, the signing algorithm, the name of the issuer (e.g., the name of Clearinghouse(s) 105), a range of dates for which the certificate is considered to be valid, the name Electronic Digital Content Store(s) 103, the public key of the Electronic Digital Content Store(s) 103, and a hash code of all of the other information signed using the private key of the Clearinghouse(s) 105. Entities that have the Public Key 621 of the Clearinghouse(s) 105 can validate the certificate and then be assured that a SC(s) with a signature that can be validated using the public key from the certificate is a valid SC(s).

After the Electronic Digital Content Store(s) 103 has received its digital certificate that was created by the Clearinghouse(s) 105 and the necessary tools for processing the SC(s) from the digital content label, it can begin offering Content 113 that can be purchased by End-User(s). The Electronic Digital Content Store(s) 103 includes its certificate and the Transaction SC(s) 640 and signs the SC(s) using its Digital Signature 643. The End-User Device(s) 109 verifies that the Electronic Digital Content Store(s) 103 is a valid distributor of Content 113 on the Secure Digital Content Electronic Distribution System 100 by first checking the digital certificate revocation list and then using the Public Key 621 of the Clearinghouse(s) 105 to verify the information in the digital certificate for the Electronic Digital Content Store(s) 103. A digital certificate revocation list is maintained by the Clearinghouse(s) 105. The revocation list may be included as one of the parts in

a License SC(s) 660 that is created by the Clearinghouse(s) 105. End-User Device(s) 109 keep a copy of the revocation list on the End-User Device(s) 109 so they can use it as part of the Electronic Digital Content Store(s) 103 digital certificate validation. Whenever the End-User Device(s) 109 receives a License SC(s) 660 it determines whether a new revocation list is included and if so, the local revocation list on the End-User Device(s) 109 is updated.

B. Rights Management Processing

Order SC(s) Analysis

The Clearinghouse(s) 105 receives an Order SC(s) 650 from an End-User(s) after the End-User(s) has received the Transaction SC(s) 640, which include the Offer SC(s) 641, from the Electronic Digital Content Store(s) 103. The Order SC(s) 650 consists of parts that contain information relative to the Content 113 and its use, information about the Electronic Digital Content Store(s) 103 that is selling the Content 113, and information about the End-User(s) that is purchasing the Content 113. Before the Clearinghouse(s) 105 begins processing the information in the Order SC(s) 650, it first performs some processing to insure that the SC(s) is in fact valid and the data it contains has not been corrupted in any way.

Validation

The Clearinghouse(s) 105 begins the validation of Order SC(s) 650 by verifying the digital signatures, then the Clearinghouse(s) 105 verifies the integrity of the Order SC(s) 650 parts. To validate the digital signatures, first the Clearinghouse(s) 105 decrypts the Contents 631 of the signature itself using the Public Key 661 of the signing entity included if signed. (The signing entity could be the Content Provider(s) 101, the Electronic Digital Content Store(s) 103, the End User Device(s) 109 or any combination of them.) Then, the Clearinghouse(s) 105 calculates the digest of the concatenated part digests of the SC(s) and compares it with the digital signature's decrypted Content 113. If the two values match, the digital signature is valid. To verify the integrity of each part, the Clearinghouse(s) 105 computes the digest of the part and compares it to the digest value in the BOM. The Clearinghouse(s) 105 follows the same process to verify the digital signatures and part integrity for the Metadata and Offer SC(s) 641 parts included within the Order SC(s) 650.

The process of verification of the Transaction and Offer SC(s) 641 digital signatures also

indirectly verifies that the Electronic Digital Content Store(s) 103 is authorized by the Secure Digital Content Electronic Distribution System 100. This is based on the fact that the Clearinghouse(s) 105 is the issuer of the certificates. Alternately, the Clearinghouse(s) 105 would be able to successfully verify the digital signatures of the Transaction SC(s) 640 and Offer SC(s) 641 using the public key from the Electronic Digital Content Store(s) 103, but only if the entity signing the SC(s) has ownership of the associated private key. Only the Electronic Digital Content Store(s) 103 has ownership of the private key. Notice that the Clearinghouse(s) 105 does not need to have a local database of the Electronic Digital Content Store(s) 103. Since the store uses the Clearinghouse Public Key to sign the Transaction SC(s) 640 Offer SC(s) 641 public keys.

Then, the Store Usage Conditions 519 of the Content 113 which the End-User(s) is purchasing are validated by the Clearinghouse(s) 105 to insure that they fall within the restrictions that were set in the Metadata SC(s) 620. Recall that the Metadata SC(s) 620 is included within the Order SC(s) 650.

Key Processing

Processing of the encrypted Symmetric Keys 623 and of the watermarking instructions are done by the Clearinghouse(s) 105 after authenticity and the integrity check of the Order SC(s) 650, the validation of the Electronic Digital Content Store(s) 103, and the validation of the Store Usage Conditions 519 have been completed successfully. The Metadata SC(s) 620 portion of the Order SC(s) 650 typically has several Symmetric Keys 623 located in the Key Description part that were encrypted using the Public Key 621 of the Clearinghouse(s) 105. Encryption of the Symmetric Keys 623 are done by the Content Provider(s) 101 when the Metadata SC(s) 620 was created.

One Symmetric Key 623 are used for decrypting the watermarking instructions and the others for decrypting the Content 113 and any encrypted metadata. Since Content 113 can represent a single song or an entire collect of songs on a CD, a different Symmetric Key 623 may be used for each song. The watermarking instructions are included within the Metadata SC(s) 620 portion in the Order SC(s) 650. The Content 113 and encrypted metadata are in the Content SC(s) 630 at a Content Hosting Site(s) 111. The URL and part names of the encrypted Content 113 and metadata parts, within the Content SC(s) 630, are included in the Key Description part of the Metadata SC(s) 620 portion of the Order SC(s) 650. The Clearinghouse(s) 105 uses its private key to decrypt the

Symmetric Keys 623 and then encrypts each of them using the Public Key 661 of the End-User Device(s) 109. The Public Key 661 of the End-User Device(s) 109 is retrieved from the Order SC(s) 650. The new encrypted Symmetric Keys 623 are included in the Key Description part of the License SC(s) 660 that the Clearinghouse(s) 105 returns to the End-User Device(s) 109.

During the time of processing the Symmetric Keys 623, the Clearinghouse(s) 105 may want to make modifications to the watermarking instructions. If this is the case, then after the Clearinghouse(s) 105 decrypts the Symmetric Keys 623, the watermarking instructions are modified and re-encrypted. The new watermarking instructions are included as one of the parts within the License SC(s) 660 that gets returned to the End-User Device(s) 109.

If all of the processing of the Order SC(s) 650 is successful, then the Clearinghouse(s) 105 returns a License SC(s) 660 to the End-User Device(s) 109. The End-User Device(s) 109 uses the License SC(s) 660 information to download the Content SC(s) 630 and access the encrypted Content 113 and metadata. The watermarking instructions are also executed by the End-User Device(s) 109.

If the Clearinghouse(s) 105 is not able to successfully process the Order SC(s) 650, then an HTML page is returned to the End-User Device(s) 109 and displayed in an Internet browser window. The HTML page indicates the reason that the Clearinghouse(s) 105 was unable to process the transaction.

In an alternate embodiment, if the user has purchased a copy of the Content 113 prior to the release date set for the sale, the License(s) SC 660 is returned without the Symmetric Keys 623. The License(s) SC 660 is returned to the Clearinghouse(s) 105 on or after the release date to receive the Symmetric Keys 623. As an example, the Content Provider(s) 101 allow users to download a new song prior to the release date for the song to enable customers to download the song and be prepared to play the song before a date set by the Content Provider(s) 101. This allows immediate opening of the Content 113 on the release date without having to content for bandwidth and download time on the release date.

C. Country Specific Parameters

Optionally, the Clearinghouse(s) 105 uses the domain name of the End-User Device(s) 109 and, whenever possible, the credit card billing address to determine the country location of the End-

User(s). If there are any restrictions for the sale of Content 113 in the country where the End-User(s) resides, then the Clearinghouse(s) 105 insures that the transaction being processed is not violating any of those restrictions before transmitting License SC(s) 660 to the End-User Device(s) 109. The Electronic Digital Content Store(s) 103 is also expected to participate in managing the distribution of Content 113 to various countries by performing the same checks as the Clearinghouse(s) 105. The Clearinghouse(s) 105 does whatever checking that it can in case the Electronic Digital Content Store(s) 103 is ignoring the country specific rules set by the Content Provider(s) 101.

D. Audit Logs and Tracking

The Clearinghouse(s) 105 maintains a Audit Logs 150 of information for each operation that is performed during Content 113 purchase transactions and report request transactions. The information can be used for a variety of purposes such as audits of the Secure Digital Content Electronic Distribution System 100, generation of reports, and data mining.

The Clearinghouse(s) 105 also maintains account balances in Billing Subsystem 182 for the Electronic Digital Content Store(s) 103. Pricing structures for the Electronic Digital Content Store(s) 103 is provided to the Clearinghouse(s) 105 by the digital content labels. This information can include things like current specials, volume discounts, and account deficit limits that need to be imposed on the Electronic Digital Content Store(s) 103. The Clearinghouse(s) 105 uses the pricing information to track the balances of the Electronic Digital Content Store(s) 103 and insure that they do not exceed their deficit limits set by the Content Provider(s) 101.

The following operations are typically logged by the Clearinghouse(s) 105:

- End-User Device(s) 109 requests for License SC(s) 660
- Credit card authorization number when the Clearinghouse(s) 105 handles the billing
- Dispersement of License SC(s) 660 to End-User Device(s) 109
- Requests for reports
- Notification from the End-User(s) that the Content SC(s) 630 and License SC(s) 660 were received and validated

The following information is typically logged by the Clearinghouse(s) 105 for a License

SC(s) 660:

- Date and time of the request
- Date and time of the purchase transaction
- Content ID of the item being purchased
- Identification of the Content Provider(s) 101
- Store Usage Conditions 519
- Watermarking instruction modifications
- Transaction ID 535 that was added by the Electronic Digital Content Store(s) 103
- Identification of the Electronic Digital Content Store(s) 103
- Identification of the End-User Device(s) 109
- End-User(s) credit card information (if the Clearinghouse(s) 105 is handling the billing)

The following information is typically logged by the Clearinghouse(s) 105 for an End-User's credit card validation:

- Date and time of the request
- Amount charged to the credit card
- Content ID of the item being purchased
- Transaction ID 535 that was added by the Electronic Digital Content Store(s) 103
- Identification of the Electronic Digital Content Store(s) 103
- Identification of the End-User(s)
- End-User(s) credit card information
- Authorization number received from the clearer of the credit card

The following information is typically logged by the Clearinghouse(s) 105 when a License SC(s) 660 is sent to an End-User Device(s) 109:

- Date and time of the request
- Content ID of the item being purchased
- Identification of Content Provider(s) 101
- Usage Conditions 517
- Transaction ID 535 that was added by the Electronic Digital Content Store(s) 103
- Identification of the Electronic Digital Content Store(s) 103

- Identification of the End-User(s)

The following information is typically logged when a report request is made:

- Date and time of the request
- Date and time the report was sent out
- Type of report being requested
- Parameters used to generate the report
- Identifier of the entity requesting the report

E. Reporting of Results

Reports are generated by the Clearinghouse(s) 105 using the information that the Clearinghouse(s) 105 logged during End-User(s) purchase transactions. Content Provider(s) 101 and Electronic Digital Content Store(s) 103 can request transaction reports from the Clearinghouse(s) 105 via a Payment Verification Interface 183 so they can reconcile their own transaction databases with the information logged by the Clearinghouse(s) 105. The Clearinghouse(s) 105 can also provide periodic reports to the Content Provider(s) 101 and Electronic Digital Content Store(s) 103.

The Clearinghouse(s) 105 defines a secure electronic interface which allows Content Provider(s) 101 and Electronic Digital Content Store(s) 103 to request and receive reports. The Report Request SC(s) includes a certificate that was assigned by the Clearinghouse(s) 105 to the entity initiating the request. The Clearinghouse(s) 105 uses the certificate and the SC's digital signature to verify that the request originated from an authorized entity. The request also includes parameters, such as time duration, that define the scope of the report. The Clearinghouse(s) 105 validates the request parameters to insure that requesters can only receive information for which they are permitted to have.

If the Clearinghouse(s) 105 determines that the Report Request SC(s) is authentic and valid, then the Clearinghouse(s) 105 generates a report and pack it into a Report SC(s) to be sent to the entity that initiated the request. Some reports may be automatically generated at defined time intervals and stored at the Clearinghouse(s) 105 so they can be immediately sent when a request is received. The format of the data included in the report is defined in a later version of this document.

F. Billing and Payment Verification

Billing of Content 113 can be handled either by the Clearinghouse(s) 105 or by the Electronic Digital Content Store(s) 103. In the case where the Clearinghouse(s) 105 handles the billing of the electronic Content 113, the Electronic Digital Content Store(s) 103 separates the End-User(s)' order into electronic goods and, if applicable, physical goods. The Electronic Digital Content Store(s) 103 then, notifies the Clearinghouse(s) 105 of the transaction, including the End-User(s)' billing information, and the total amount that needs to be authorized. The Clearinghouse(s) 105 authorizes the End-User(s)' credit card and returns a notification back to the Electronic Digital Content Store(s) 103. At the same time the Clearinghouse(s) 105 is authorizing the End-User(s)' credit card, the Electronic Digital Content Store(s) 103 can charge the End-User(s)' credit card for any physical goods that are being purchased. After each electronic item is downloaded by the End-User Device(s) 109, the Clearinghouse(s) 105 is notified so the End-User(s)' credit card can be charged. This occurs as the last step by the End-User Device(s) 109 before the Content 113 is enabled for use at the End-User Device(s) 109.

In the case where the Electronic Digital Content Store(s) 103 handles the billing of the electronic Content 113, the Clearinghouse(s) 105 is not notified about the transaction until the End-User Device(s) 109 sends the Order SC(s) 650 to the Clearinghouse(s) 105. The Clearinghouse(s) 105 is still notified by the End-User Device(s) 109 after each electronic item is downloaded. When the Clearinghouse(s) 105 is notified it sends a notification to the Electronic Digital Content Store(s) 103 so that the Electronic Digital Content Store(s) 103 can charge the End-User(s)' credit card.

G. Retransmissions

The Secure Digital Content Electronic Distribution System 100 provides the ability to handle retransmissions of Content 113. This is typically performed by a Customer Service Interface 184. Electronic Digital Content Store(s) 103 provides a user interface that the End-User(s) can step through in order to initiate a retransmission. The End-User(s) goes to the Electronic Digital Content Store(s) 103 site where the Content 113 item was purchased in order to request a retransmission of the Content 113.

Retransmissions of Content 113 are done when an End-User(s) requests a new copy of a

previously purchased Content 113 item because the Content 113 could not be downloaded or the Content 113 that was downloaded is not usable. The Electronic Digital Content Store(s) 103 determines whether the End-User(s) is entitled to do a retransmission of the Content 113. If the End-User(s) is entitled to a retransmission, then the Electronic Digital Content Store(s) 103 builds a Transaction SC(s) 640 that includes the Offer SC(s) 641 of the Content 113 item(s) being retransmitted. The Transaction SC(s) 640 is sent to the End-User Device(s) 109 and the identical steps as for a purchase transaction are performed by the End-User(s). If the End-User Device(s) 109 has a scrambled key(s) in the key library for the Content 113 item(s) undergoing retransmission, then the Transaction SC(s) 640 includes information that instructs the End-User Device(s) 109 to delete the scrambled key(s).

In the case where the Clearinghouse(s) 105 handles the financial settlement of Content 113 purchases, the Electronic Digital Content Store(s) 103 includes a flag in the Transaction SC(s) 640 that is carried forward to the Clearinghouse(s) 105 in the Order SC(s) 650. The Clearinghouse(s) 105 interprets the flag in the Order SC(s) 650 and proceed with the transaction without charging the End-User(s) for the purchase of the Content 113.

VIII. CONTENT PROVIDER

A. Overview

The Content Provider(s) 101 in the Secure Digital Content Electronic Distribution System 100 is the digital content label or the entity who owns the rights to the Content 113. The role of the Content Provider(s) 101 is to prepare the Content 113 for distribution and make information about the Content 113 available to Electronic Digital Content Store(s) 103 or retailers of the downloadable electronic versions of the Content 113. To provide the utmost security and rights control to the Content Provider(s) 101, a series of tools are provided to enable the Content Provider(s) 101 to prepare and securely package their Content 113 into SC(s) at their premises so that the Content 113 is secure when it leaves the Content Provider(s)' 101 domain and never exposed or accessible by unauthorized parties. This allows Content 113 to be freely distributed throughout a non-secure network, such as the Internet, without fear of exposure to hackers or unauthorized parties.

The end goal of the tools for the Content Provider(s) 101 is to prepare and package a Content

113 such as a song or series of songs into Content SC(s) 630 and to package information describing the song, approved uses of the song (content Usage Conditions 517), and promotional information for the song into a Metadata SC(s) 620. To accomplish this, the following set of tools are provided:

- Work Flow Manager 154 - Schedules processing activities and manages the required synchronization of processes.
- Content Processing Tools 155 - A collection of tools to control Content 113 file preparation including Watermarking, Preprocessing (for an audio example any required equalization, dynamics adjustment, or re-sampling) encoding and compression.
- Metadata Assimilation and Entry Tool 161 - A collection of tools used to gather Content 113 description information from the Database 160 of the Content Provider(s) and/or third party database or data import files and/or via operator interaction and provides means for specifying content Usage Conditions 517. Also provided is an interface for capturing or extracting content such as digital audio content for CDS or DDP files. A Quality Control Tool enables to preview of prepared content and metadata. Any corrections needed to the metadata or resubmission of the content for further processing can be conducted.
- SC(s) Packer Tool 152 - Encrypts and packages all Content 113 and information and calls the SC(s) Packer to pack into SC(s).
- Content Dispersement Tool (not shown) - Disperses SC(s) to designated distribution centers, such as Content Hosting Site(s) 111 and Electronic Digital Content Store(s) 103.
- Content Promotions Web Site 156 - stores Metadata SC(s) 620 and optionally additional promotional material for download by authorized Electronic Digital Content Store(s) 103.

B. Work Flow Manager 154

The purpose of this tool is to schedule, track, and manage Content 113 processing activities. This application enables multi-user access as well as allowing scheduling of Content 113 and status checking from remote locations within the Intranet or extranet of the Content Provider(s) 101. This design also allows for collaborative processing where multiple individuals can be working on multiple pieces of Content 113 in parallel and different individuals can be assigned specific responsibilities and these individuals can be spread throughout the world.

Turning now to FIG. 8 is a block diagram of the major processes of the Work Flow Manager 154 corresponding to FIG. 7. The major processes in FIG. 8 summarizes the Content 113 processing functions provided by the tools described in this section. The Work Flow Manager 154 is responsible for feeding jobs to these processes and directing jobs to the next required process upon completion of its current process. This is accomplished through a series of Application Programming Interfaces (APIs) which each processing tool calls to:

- retrieve the next job to process
- indicate successful completion of a process
- indicate unsuccessful completion of a process and reason for the failure
- provide interim status of a process (to allow initiation of processes that require only partial completion of a dependent process)
- add comments to a product which are made available to the designated processes

The Work Flow Manager 154 also has a user interface, an example Work Flow Manager User Interface 700 is illustrated in FIG. 7 which provides the following functions:

- a configuration panel to allow specification of default values and conditions to be assigned and performed during various stages of processing
- customization of the work flow rules and automated processing flows
- job scheduling
- status queries and reports
- add comments or instructions for a job associated to one or more processes
- job management (i.e. suspend, release, remove, change priority (order of processing))

Each process has a queue associated with it managed by the Work Flow Manager 154. All processes requesting jobs from the Work Flow Manager 154 results in the Work Flow Manager 154 either suspending the process (tool) in a wait state if there are no jobs currently in its associated queue or returning to the process all information about the job needed to perform its respective process. If a process is suspended in a wait state, it resumes processing when a job is placed on its queue by the Work Flow Manager 154.

The Work Flow Manager 154 also manages the flow or order of processing based on a set of defined rules. These rules can be customized by the Content Provider(s) 101 if it has special

processing requirements or configures specific defaults rules. When a process reports completion of its assigned task, it notifies the Work Flow Manager 154 of this status and the Work Flow Manager 154 decides what queue the job gets placed on next based on the defined rules.

Comments indicating special handling instructions or notices may also be attached to the product at any of the processing steps via either the programming API or manually through the Work Flow Manager User Interface 700 or processor interfaces.

The processes in the Work Flow Manager 154 are implemented in Java in the preferred embodiment but other programming languages such as C/C++, Assembler and equivalent can be used. It should be understood that the processes described below for the Work Flow Manager 154 can run on a variety of hardware and software platforms. The Work Flow Manager 154 as a complete system or as any of it's constitute processes may be distributed as an application program in a computer readable medium including but not limited to electronic distribution such as the web or on floppy diskettes, CD ROMS and removable hard disk drives.

Turning now to FIG. 8 is a block diagram of the major processes of the Work Flow Manager 154 corresponding to FIG. 7. The following sections summarize each process and describes the information or action required by each process.

1. Products Awaiting Action/Information Process 801

Jobs are placed on specific processes queues once all information required by that process is available and the job has already successfully completed all dependent processing. A special queue exists in the Work Flow Manager 154 which is used to hold jobs that are not currently available for processing due to missing information or a failure that prevent further processing. These jobs are placed in the Products Awaiting Action/Information Process 801 queue. Each job in this queue has associated status to indicate the action or information it is waiting on, the last process that worked on this job, and the next process(es) this job is queued to once the missing or additional information is provided or the required action is successfully completed.

Completion of any process causes the Work Flow Manager 154 to check this queue and determine if any job in this queue was awaiting the completion of this process (action) or information provided by this process. If so, that job is queued to the appropriate process queue.

2. New Content Request Process 802

The Content Provider(s) 101 determines those products (for example, a product may be a song or a collection of songs) it wishes to sell and deliver electronically. The initial function of the Work Flow Manager 154 is to enable an operator to identify these products and to place them on the queue of the New Content Request Process 802. The Content Provider(s) 101 may specify through configuration options, what information is prompted for on the product selection interface. Enough information is entered to uniquely identify the product. Optionally, additional fields may be included to request manual entry of the information required to initiate the audio processing phase in parallel with the metadata acquisition. If not provided manually, this information can optionally be retrieved from default configuration settings or from the Database 160 of the Content Provider(s), obtained in the first stage of Metadata Processing as in Automatic Metadata Acquisition Process 803. The makeup and capabilities of the Content 113 in the Database 160 of the Content Provider(s) determines the Content selection process.

If the required information needed to perform a query to the Database 160 of the Content Provider(s) 101 is specified, the job is processed by the Automatic Metadata Acquisition Process 803. In a music embodiment, to properly schedule the product for audio processing, the product's genre and the desired compression levels are specified as well as the audio PCM or WAV filename(s). This information may be entered as part of the product selection process or selected via a customized query interface or Web browser function. Specification of this information enables the product to be scheduled for content processing.

The product selection user interface provides an option enabling the operator to specify whether the product can be released for processing or whether it are held pending further information entry. If held, the job is added to the queue of the New Content Request Process 802 awaiting further action to complete data entry and/or release the product for processing. Once the product is released, the Work Flow Manager 154 evaluates the information specified and determines which processes the job is ready to be passed to.

If adequate information is provided to enable an automated query to the Database 160 of the Content Provider(s)' 101, the job is queued for Automatic Metadata Acquisition Process 803. If the database mapping table has not been configured for the Automatic Metadata Acquisition Process

803, the job is queued for Manual Metadata Entry Process 804 (see Automatic Metadata Acquisition Process 803 section for details on the Database Mapping Table).

If the required general information for audio processing and the specific information required for watermarking is specified, the job is queued for Watermarking Process 808 (the first phase of content processing). If any of the required information is missing when the job is released, the job is queued to the queue of the Products Awaiting Action/Information Process 801 along with status indicating the information that is missing.

If the status indicates that the filename of the Content 113, for example where the Content 113 is audio and the PCM or WAV file is missing, this may indicate that a capture (or digital extraction from digital media) is required. The audio processing functions require that the song files be accessible via a standard file system interface. If the songs are located on external media or a file system that is not directly accessible to the audio processing tools, the files are first be copied to an accessible file system. If the songs are in digital format but on CD or Digital Tape, they are extracted to a file system accessible to the audio processing tools. Once the files are accessible, the Work Flow Manager User Interface 700 is used to specify or select the path and filename for the job so that it can be released to the watermarking process, assuming all other information required for watermarking has also been specified.

3. Automatic Metadata Acquisition Process 803

The Automatic Metadata Acquisition Process 803 performs a series of queries to the Database 160 of the Content Provider(s) 101 or a staging database where data has been imported, in an attempt to obtain as much of the product information as possible in an automated fashion. The Automatic Metadata Acquisition Process 803 requires the following information prior to allowing items to be placed on its queue:

- database mapping table with adequate information to generate queries to the Database 160 of the Content Provider(s) 101
- product information required to perform queries
- adequate product information to uniquely define product

An automated query is performed to the Database 160 of the Content Provider(s) 101 to

obtain the information necessary to process this Content 113. For example, if the Content 113 is music, the information needed to perform this query could be the album name or may be a UPC or a specific album or selection ID as defined by the Content Provider(s) 101. Of the information to be obtained, some is designated as required (see the section on Automatic Metadata Acquisition Process 803 for details). If all required information is obtained, the job is next queued for Usage Conditions Process 805. If any required information is missing, the song is queued for Manual Metadata Entry Process 804. If any jobs in the Products Awaiting Action/Information Process 801 queue are waiting for any of the information obtained in this step, the jobs status is updated to indicate that it is no longer waiting for this information. If that job no longer has any outstanding requirements, it is queued to the next defined queue.

4. Manual Metadata Entry Process 804

The Manual Metadata Entry Process 804 provides a means for an operator to enter missing information. It has no dependencies. Once all required information is specified, the job is queued for Usage Conditions Process 805.

5. Usage Conditions Process 805

The Usage Conditions Process 805 allows specification of product uses and restrictions. The Usage Conditions Process 805 may require some metadata. Upon completion of Usage Conditions specifications, the job is eligible to be queued for Metadata SC(s) Creation Process 807 unless the Supervised Release Process 806 option has been requested or is configured as the default in the Work Flow Manager 154 rules. In that case, the job is queued for Supervised Release Process 806. Before queuing to Metadata SC(s) Creation Process 807, the Work Flow Manager 154 will first assure that all dependencies for that process have been met (see below). If not, the job is queued to the Products Awaiting Action/Information Process 801.

6. Supervised Release Process 806

The Supervised Release Process 806 allows a quality check and validation of information specified for the digital content product. It does not have any dependencies. Comments previously

attached to the job at any stage of the processing for this product can be reviewed by the Supervisor and appropriate action taken. After reviewing all information and comments, the Supervisor has the following options:

- approve release and queue the product for Metadata SC(s) Creation Process 807
- modify and/or add information and queue the product for Metadata SC(s) Creation Process 807
- add comments to the job and re-queue for Manual Metadata Entry Process 804
- add comments and queue the job to the queue for Products Awaiting Action/Information Process 801

7. Metadata SC(s) Creation Process 807

The Metadata SC(s) Creation Process 807 gathers together all the information collected above as well as other information required for the Metadata SC(s) 620 and calls the SC(s) Packer Process to create the Metadata SC(s) 620. This tool requires the following as input:

- the required metadata
- the usage conditions
- the encryption keys used in the encryption stage of all quality levels for this product

This last dependency requires that the associated audio objects completed the audio processing phase before the Metadata SC(s) 620 can be created. Upon completion of the Metadata SC(s) Creation Process 807, the job is queued to either the queue for Final Quality Assurance Process 813 or Content Dispersement Process 814 based on defined work flow rules.

8. Watermarking Process 808

The Watermarking Process 808 adds copyright and other information to the Content 113. For an embodiment where the Content 113 is a song, this tool requires the following as input:

- song filename(s) (multiple filenames if album)
- watermarking instructions
- watermarking parameters (information to be included in the watermark)

Upon completion of the Watermarking Process 808, the job is queued for Preprocessing and

Compression Process 809 if its required input is available or otherwise queued to the Products Awaiting Action/Information Process 801.

9. Preprocessing and Compression Process 809

The Preprocessing and Compression Process 809 encodes the Content 113 to the specified compression level performing any required preprocessing first. Queuing a job to this queue actually create multiple queue entries. A job is created for each compression level of the product desired. The encoding processes can be performed in parallel on multiple systems. This tool requires the following input:

- watermarked content filename(s) (multiple filenames if Content 113 is an album)
- quality levels for product (could be preconfigured)
- compression algorithm (could be preconfigured)
- product genre (if required by preprocessor)

Upon completion of the encoding process, the jobs are queued to the Content Quality Control Process 810 if configured by the work flow rules. If not, the jobs are queued for Encryption Process 811.

If third party providers of encoding tools do not provide a method to display the percentage of the Content 113, such as audio, that has been processed or a method to indicate the amount of Content 113 that has been encoded as a percentage of the entire selection of Content 113 selected, in FIG. 11 there is shown a flow diagram 1100 of a method to determine the encoding rate of Digital Content for the Content Preprocessing and Compression tool of FIG. 8. The method begins with the selection of the desired encoding algorithm and a bit rate, step 1101. Next, a query is made to determine if this algorithm and encoding rate has a previously calculated rate factor, step 1102. The rate factor is the factor used to determine the rate of compression for a specific encoding algorithm and a specific bit rate. If no previously calculated rate factor is stored, a sample of the Content 113 is encoded for a predetermined amount of time. The predetermined period of time in the preferred embodiment is a few seconds. This rate of encoding for a predetermined period of time is used to calculate a new rate factor R_{NEW}. Calculating a new rate factor R_{NEW} knowing the amount of time and the amount of Content 113 encoded is R_{NEW} = (length of Digital Content encoded)/(amount of

time), step 1108. The Content 113 is encoded and the encoding status is displayed using the previously calculate rate factor R_{NEW}, step 1109. This encoding rate factor R_{NEW} is then stored, step 1107, for future use for this encoding algorithm and encoding bit rate. If the selected algorithm has a previously calculated rate factor R_{STORED}, step 1103. The Content 113 is encoded and the progression displayed using the previously calculated rate factor R_{STORED}, step 1104. In the meantime, a current rate factor, $R_{current}$ is calculated for this selected algorithm and bit rate, step 1105. This current rate factor $R_{current}$ is used to update the stored rate factor R_{NEW} = AVERAGE OF (R_{STORED} + $R_{CURRENT}$), step 1106. The iterative update of the rate factor enables the determination of the encoding rate to become more and more accurate with each subsequent use for a particular encoding algorithm and bit rate. The new rate R_{NEW} is then stored for future use, step 1107. The updating of R_{STORED} may not be made if the current rate factor $R_{current}$ is out range for the previously stored rate factor R_{STORED} by a given range or threshold.

The display of the encoding status can then be presented. The encoding status includes along with the current encoding rate, the display of the percentage of the total Content 113 displayed as a progression bar based on the encoding rate and the total length of the file for the Content 113. The encoding status can also include the time remaining for the encoding. The time remaining for the encoding can be calculated by dividing the encoding rate calculated $R_{CURRENT}$ by the total length of the file for Content 113. The encoding status can be transferred to another program that may invoke the calling process. This can help supervisory programs to encoding or co-dependent programs on encoding be operated and be batched for processing more efficiently. It should be understood, in an alternative embodiment, that encoding can include the step of watermarking.

10. Content Quality Control Process 810

The Content Quality Control Process 810 is similar in function to the Supervised Release Process 806. It is an optional step allowing someone to validate the quality of the content processing performed thus far. This has no dependencies other than completion of the Watermarking Process 808 and the encoding portion of the Preprocessing and Compression Process 809. Upon completion of the Content Quality Control Process 810 the following options are available:

- the jobs can be released and queued for Encryption Process 811.

- comments can be attached and one or more of the jobs re-queued for Preprocessing and Compression Process 809.

The last option requires that the unencoded watermarked version of the song file remain available until after Content Quality Control Process 810.

11. Encryption Process 811

The Encryption Process 811 calls the appropriate Secure Digital Content Electronic Distribution Rights Management function to encrypt each of the watermarked/encoded song files. This process has no dependencies other than completion of all other audio processing. Upon completion of the Encryption Process 811 process, the job is queued for Content SC(s) Creation Process 812.

12. Content SC(s) Creation Process 812

The Content SC(s) Creation Process 812 Process may require some metadata files to be included in the Content SC(s) 630. If files other than the Content 113 are required, the files are gathered and the SC(s) Packer Process is called to create a Content SC(s) 630 for each compression level of the Content 113 (e.g. a song) created. Upon completion of the Content SC(s) Creation Process 812, the song is queued to either the Final Quality Assurance Process 813 or Content Dispersement Process 814 queue based on defined work flow rules.

13. Final Quality Assurance Process 813

Final Quality Assurance Process 813 is an optional step that allows a cross reference check between the associated Metadata and Content SC(s) 630 to verify that they match up correctly and that all information and Content 113 contained therein are correct. Upon completion of Final Quality Assurance Process 813, the jobs are queued for Content Dispersement Process 814. If a problem is found, the job in most cases has to be re-queued to the failing stage. Rework at this stage is much more costly since the product has to go through re-encryption and repacking in addition to the reprocessing required to correct the problem. It is highly recommended that the prior assurance stages be used to assure the quality of the Content 113 and accuracy and completeness of the

information.

14. Content Dispersement Process 814

The Content Dispersement Process 814 Process is responsible for transferring the SC(s) to the appropriate hosting sites. After the successful transfer of the SC(s), the job completion status is logged and the job is deleted from the queue. If a problem occurs in transferring the SC(s), after a defined number of retries, the job is flagged in the Workflow Manager Tool 154 as having failed along with the error encountered.

15. Work Flow Rules

The Work Flow Rules for FIG. 8 operate in three major systems as follows:

A: Work Flow Manager Tool 154

1. New Content Request Process 802
2. Products Awaiting Action/Information Process 801
3. Final Quality Assurance Process 813
4. Content Dispersement (and Notification) Process 814

B: Metadata Assimilation and Entry Tool 161

1. Automatic Metadata Acquisition Process 803
2. Manual Metadata Entry Process 804
3. Supervised Release Process 806
4. Metadata SC(s) Creation Process 807

C: Content Processing Tools 155

1. Watermarking Process 808 (requires copyright data)
2. Preprocessing and Compression Process 809
3. Content Quality Control Process 810
4. Encryption Process 811

5. Content SC(s) Creation Process 812

Work Flow

The Content 113 selection operator inputs a new product and it starts out queued onto **A1** (New Content Request Process 802).

A1: When the Content 113 selection operator releases it to the Work Flow Manager Tool 154, then it gets queued onto **B1** (the Automatic Metadata Acquisition Process 803).

A2: coming from step **B1** (the Automatic Metadata Acquisition Process 803),
or step **B2** (Manual Metadata Entry Process 804),
or step **B3** (Supervised Release Process 806)
on its way to step **Before** (the Metadata SC(s) Creation Process 807)
[needs the encryption keys].

coming from step **Before** (the Metadata SC(s) Creation Process 807)
on its way to either step **A3** (the Final Quality Assurance Process 813) or step **A4** (the Content Dispersement Process 814)
[needs the Content SC(s) 630].

coming from step **C1** (the Watermarking Process 808)
on its way to step **C2** (the Preprocessing and Compression Process 809)
[needs the metadata for Preprocessing and Compression Process 809].

coming from step **C4** (the Encryption Process 811)
on its way to step **C5** (the Content SC(s) Creation Process 812)
[needs the metadata for Content SC(s) 630 Packing].

coming from step **C5** (the Content SC(s) Creation Process 812)
on its way to either step **A3** (the Final Quality Assurance Process 813) or step **A4** (the Content Dispersement Process 814)
[needs the Metadata SC(s) 620].

A3: After step A3 (the Final Quality Assurance Process 813),
place onto queue **B2** (Manual Metadata Entry Process 804),
or place onto queue **B3** (Supervised Release Process 806),

or place into queue as required by the quality assurance operator.

A4: After step **A4** (Content Dispersement Process 814),

the Work Flow Manager Tool 154 is done for this product.

B1: After step **B1** (the Automatic Metadata Acquisition Process 803),

if the metadata needed for step **C1** (the Watermarking Process 808) is present, then place an entry representing this product onto queue **C1.**

(do the following logic also)

if either 1- any required metadata is missing, or 2- there are comments directed to the manual metadata providers, then also place the product onto queue **B2** (Manual Metadata Entry Process 804),

else if supervised release was requested for this product, then place the product onto queue **B3** (Supervised Release Process 806).

else if the product has all the information from the Content Processing Tools 155 for all of the requested quality levels, then place the product onto queue **Before** (the Metadata SC(s) Creation Process 807),

else flag the product as needs the encryption keys and place the product onto queue **A2** (Products Awaiting Action/Information Process 801).

B2: During step **B2** (Manual Metadata Entry Process 804),

if step **C1** (the Watermarking Process 808) has not been done and the metadata needed for step **C1** is present, then place an entry representing this product onto queue **C1.**

(do the following logic also)

if metadata needed for step **C2** (the Preprocessing and Compression Process 809) just been provided, then

(do the following logic also)

if all the metadata that can be gathered by the Metadata Assimilation and Entry Tool 161 is present, then

if supervised release was requested for this product, then place the product onto queue **B3** (Supervised Release Process 806)

else

if all the information from step **C4** (the Encryption Process 811) of the Content Processing Tools 155 is present, then place this product onto queue **Before** (the Metadata SC(s) Creation Process 807)

else flag the product as needs the encryption keys and place this product onto queue **A2** (Products Awaiting Action/Information Process 801).

else

if the metadata provider requested a forced supervised release, then place the product onto queue **B3** (Supervised Release Process 806)

else do nothing (keep the product on queue **B2** (Manual Metadata Entry Process 804)).

B3: During step **B3** (Supervised Release Process 806),

if this operator is sending the product back to step **B2** (Manual Metadata Entry Process 804), then place the product on queue **B2**.

else if this operator released the product, then

if all the information from step **C4** (the Encryption Process 811) of the Content Processing Tools 155 is present, then place this product onto queue **Before** (the Metadata SC(s) Creation Process)

else flag the product as needs the encryption keys and place this product onto queue **A2** (Products Awaiting Action/Information Process 801).

else the product remains on queue **B3** (Supervised Release Process 806).

Before: After step **Before** (the Metadata SC(s) Creation Process 807),

flag the product Metadata has been packed .

if all the (product/quality level) tuples have been packed, then

if the Content Provider(s)' 101 configuration specifies Quality Assure the SC(s), then place this product onto queue **A3** (the Final Quality Assurance Process 813)

else place this product onto queue **A4** (the Content Dispersement Process 814).

else flag the product as needs the Content 113 SC(s) and place this product onto queue **A2** (Products Awaiting Action/Information Process 801).

C1: After step **C1** (the Watermarking Process 808),

if the metadata needed for step **C2** (the Preprocessing and Compression Process 809) is present, then create an entry for each (product/quality level) tuple and place them onto queue **C2**,

else flag the product as needs the metadata for Preprocessing/Compression and place this product onto queue **A2** (Products Awaiting Action/Information Process 801).

C2: After step **C2** (the Preprocessing and Compression Process 809),

if the Content Provider(s)' 101 configuration specifies Content Quality Control Process 810 , then place this (product/quality level) tuple onto queue **C3** (the Content Quality Control Process 810),

else place this (product/quality level) tuple onto queue **C4** (the Encryption Process 811).

C3: After step **C3** (the Content Quality Control Process 810), then place this (product/quality level) tuple onto queue **C4** (the Encryption Process 811).

C4: After step **C4** (the Encryption Process 811),

provide the needed information (i.e., the Symmetric Key 623 generated by the Process and used to encipher the Content 113) to the Metadata Assimilation and Entry Tool 161.

if all the metadata that's required for the Content SC(s) 630 is present, then place this (product/quality level) tuple onto queue **C5** (the Content SC(s) Creation Process 812),

else flag the product as needs the metadata for Content SC(s) 630 Packing and place this (product/quality level) tuple onto **A2** (Products Awaiting Action/Information Process 801).

C5: After step **C5** (the Content SC(s) Creation Process 812),

flag the quality level the Content 113 at this quality level has been packed .

if all the (product/quality level) tuples have been packed, then

if the product is flagged Metadata has been packed , then

if the Content Provider(s)' 101 configuration specifies Quality Assure the SC(s), then place this product onto queue **A3** (the Final Quality Assurance Process 813)

else place this product onto queue **A4** (the Content Dispersement Process 814)

else flag the product as needs the Metadata SC(s) 620 and place this product onto queue **A2** (Products Awaiting Action/Information Process 801).

else (all the (product/quality level) tuples have not been packed) do nothing (another (product/quality level) tuple triggers an action).

C. Metadata Assimilation and Entry Tool

Metadata consists of the data describing the Content 113 for example in music, title of the recording, artist, author/composer, producer and length of recording. The following description is based upon Content 113 being music but it should be understood by those skilled in the art that other content types e.g., video, programs, multimedia, movies, and equivalent, are within the true scope and meaning of the present invention.

This Subsystem brings together the data the Content Provider(s) 101 provides to the Electronic Digital Content Store(s) 103 to help promote the sale of the product (e.g., for music, sample clips by this artist, history of this artist, list of albums on which this recording appears, genres associated with this artist and/or product), the data the Content Provider(s) 101 provides to the End-User(s) with the purchased product (e.g., artist, producer, album cover, track length), and the different purchase options (the Usage Conditions 517) the Content Provider(s) 101 wants to offer the End-User(s). The data is packaged into a Metadata SC(s) 620 and made available to the Electronic Digital Content Store(s) 103. To accomplish this, the following tools are provided:

- Automatic Metadata Acquisition Tool
- Manual Metadata Entry Tool
- Usage Conditions Tool
- Supervised Release Tool

These tools enable Content Provider(s) 101 to implement the processes described above for Work Flow Manager 154. Tools described here are a toolkit based on Java in the preferred embodiment but other programming languages such as C/C++, Assembler and equivalent can be used.

1. Automatic Metadata Acquisition Tool

The Automatic Metadata Acquisition Tool provides a user the ability to implement the Automatic Metadata Acquisition Process 803 described above. The Automatic Metadata Acquisition Tool is used to access the Database 160 of the Content Provider(s) 101 and to retrieve as much data as possible without operator assistance. Configuration methods are available to automate this process. The Content Provider(s) 101 can tailor the default metadata template to identify the types of data this Content Provider(s) 101 wants to provide to End-User(s) (e.g., composer, producer, sidemen, track length) and the types of promotional data the Content Provider(s) 101 provides to the Electronic Digital Content Store(s) 103 (e.g., for a music example, sample clips by this artist, a history of this artist, the list of albums on which this recording appears, genres associated with this artist). The default metadata template includes data fields which are required by the End-User Device(s) 109, data fields which can be optionally provided to the End-User Device(s) 109 and a sample set of data fields, targeted to the Electronic Digital Content Store(s) 103, that promote the artist, album, and/or single.

To extract the template data fields from the Database 160 of the Content Provider(s) 101 the Automatic Metadata Acquisition Tool uses a table that maps the type of data (e.g., composer, producer, a biography of the artist) to the location within the database where the data can be found. Each of the Content Provider(s) 101 help specify that mapping table for their environment.

The Automatic Metadata Acquisition Tool uses a metadata template of the Content Provider(s) 101 and mapping table to acquire whatever data is available from the Databases 160 of the Content Provider(s) 101. The status of each product is updated with the result of the Automatic Metadata Acquisition Process 803. A product which is missing any required data is queued for Manual Metadata Entry Process 804, otherwise it is available for packing into a Metadata SC(s) 620.

2. Manual Metadata Entry Tool

The Manual Metadata Entry Tool provides a user the ability to implement the Manual Metadata Entry Process 804 described above. The Manual Metadata Entry Tool allows any properly authorized operator to provide the missing data. If the operator determines that the missing data is unavailable, the operator can attach a comment to the product and request supervised release. The

Content Provider(s) 101 may require, for quality assurance reasons, that the product undergo supervised release. Once all the required data is present, and if supervised release has not been requested, then the product is available for packing into a Metadata SC(s) 620.

3. Usage Conditions Tool

The Usage Conditions Tool provides a user the ability to implement the Usage Conditions Process 805 described above. The process of offering Content 113 for sale or rent (limited use), using electronic delivery, involves a series of business decisions. The Content Provider(s) 101 decides at which compression level(s) the Content 113 is made available. Then for each compressed encoded version of the Content 113, one or more usage conditions are specified. Each usage condition defines the rights of the End-User(s), and any restrictions on the End-User(s), with regard to the use of the Content 113.

As part of Content Processing Tools 155, a set of usage conditions (End-User(s) rights and restrictions) is attached to the product.

A usage condition defines:

1. the compression encoded version of the Content 113 to which this usage condition applies.
2. the type of user covered by this usage condition (e.g., business, private consumer)
3. whether this usage condition allows for the purchase or the rental of the Content 113.

   For a rental transaction:

   - the measurement unit which is used to limit the term of the rental (e.g., days, plays).
   - the number of the above units after which the Content 113 will no longer play.

   For a purchase transaction:

   - the number of playable copies the End-User(s) is allowed to make.
   - onto what kinds of media can he/she make those copies (e.g., CD-Recordable (CD-R), MiniDisc, Personal Computer).

4. the period of time during which the purchase/rental transaction is allowed to occur (i.e., an End-User(s) can purchase/rent under the terms of this usage condition only after the beginning availability date and before the last date of availability).

5. the countries from which an End-User(s) can transact this purchase (or rental).
6. the price of the purchase/rental transaction under this usage condition
7. the watermarking parameters.
8. the types of events which require notification of the Clearinghouse(s) 105.

An Example of a Set of Usage Conditions

The Content Provider(s) 101 may decide to test the North American market's acceptance to the re-release of the children's song by a popular children's vocalist during the fourth quarter 1997. The test will make the song available in two different compression encoding versions: 384Kbps and 56Kbps. The 384Kbps version can be bought (and one copy made onto MiniDisc) or rented (for two weeks), while the 56Kbps version can only be bought (and no copies made). The watermarking instructions is the same for any purchase/rental, and the Content Provider(s) 101 wants the Clearinghouse(s) 105 to count every copy made. This would create Usage Conditions as follows:

	Usage Condition 1	Usage Condition 2	Usage Condition 3
compressed encoded version	384Kbps	384Kbs	56Kbps
type of user	private consumer	private consumer	private consumer
type of transaction	purchase	rental	purchase
availability dates	1 Oct 1997 - 31 Dec 1997	1 Oct 1997 - 31 Dec 1997	1 Oct 1997 - 31 Dec 1997
countries	USA and Canada	USA and Canada	USA and Canada
watermarking	std.	std.	std.
notifying events	copy action	none	none
number of copies	1	0	0
onto what media	MiniDisc	not applicable	not applicable
term of rental	not applicable	14 days	not applicable
price	Price 1	Price 2	Price 3

4. Parts of the Metadata SC(s) 620

Below are some of the kinds of data that the Metadata Assimilation and Entry Tool 161 gathers for inclusion into the Metadata SC(s) 620. An attempt has been made to group the data into SC(s) parts by function and destination.

product ID [src:content provider;]
[dest: everybody;]
licensor label company [dest: EMS; end-user;]

licensee label company [dest: EMS; end-user;]
source (publisher) of this object (sublicensee label company) [dest: everybody;]
type of object (i.e., a single object or an array of objects)
object ID [dest: everybody;]
International Standard Recording Code (ISRC)
International Standard Music Number (ISMN)

usage conditions (src: content provider; dest: EMS, end-user, Clearinghouse(s) 105)
purchased usage conditions (src: EMS; dest: end-user, Clearinghouse(s) 105)
the set of usage conditions (consumer restrictions and rights) for the use of the object (sound recording)
an individual entry in the array of usage conditions
the compression encoded version of the Content 113 to which this usage condition applies
whether this usage condition allows for the purchase or the rental of the Content 113
for a rental transaction:
the measurement unit which is used to limit the term of the rental (e.g., days, plays).
the number of the above units after which the Content 113 will no longer play.
for a purchase transaction:
the number of playable copies the End-User(s) is allowed to make.
onto what kinds of media can (s)he make those copies (e.g., CD-Recordable (CD-R), MiniDisc, personal computer).
the period of time during which the purchase/rental transaction is allowed to occur (i.e., an End-User(s) can purchase/rent under the terms of this usage condition only after the beginning availability date and before the last date of availability)
a pointer to the countries from which an End-User(s) can transact this purchase (or rental)
the price of the purchase/rental transaction under this usage condition

a pointer to the encrypted watermarking instructions and parameters

a pointer to the types of events which require notification of the Clearinghouse(s) 105

purchase data (encrypted; optional info; src: EMS; dest: end-user, Clearinghouse(s) 105)

purchase date

purchase price

bill to name and address

consumer name and address

country of the consumer (best guess)

metadata 1 (src: content provider; dest: EMS, end-user)

an array of {

copyright information

for the composition

for the sound recording

title of song

principal artist(s)

}

a pointer to {

the artwork (e.g., album cover);

the format of the artwork (e.g., GIF, JPEG);

}

optional info:

an array of additional information {

composer

publisher

producer

sidemen
date of recording
date of release
lyrics
track name (description) / track length
list of albums on which this recording appears
genre(s)
}

metadata 2 (src: content provider; dest: EMS)
an array of structures, each representing different quality levels of the same sound recording
{
the sound recording;
the quality level of the sound recording;
the size (in bytes) of the (probably compressed) sound recording;
}

metadata 3 (src: content provider; dest: EMS, end-user)
optional info:

promotional material:
a pointer to artist promotion material {
a URL to the artist's web site;
background description(s) of the artist(s);
artist-related interviews (along with format of the interview (e.g., text, audio, video));
reviews (along with format of the reviews (e.g., text, audio, video));
sample clips (and its format and compression level);
recent and upcoming concerts/appearances/events - their dates and locations;

}

a pointer to album promotion material {

sample clip (and its format and compression level);

background description(s) of the producer, and/or the composer, and/or the movie/play/cast, and/or the making of the album, etc.;

non-artist-related interviews (along with format of the interview (e.g., text, audio, video));

reviews (along with format of the reviews (e.g., text, audio, video));

genre(s);

}

single promotions:

sample clip (and its format and compression level)

background description(s) of the producer, and/or the composer, and/or the movie/play/cast, and/or the making of the single, etc.

reviews (along with format of the reviews (e.g., text, audio, video))

5. Supervised Release Tool

Supervised Release Tool provides a user the ability to implement the Supervised Release Process 806 described above. An individual designated by the Content Provider(s) 101 as having supervised release authority, may call up a product awaiting supervised release (i.e., a product on the queue of the Supervised Release Process 806), examine its Contents 113 and its accompanying comments, and either

approve its Contents 113 and release the product for packing into a Metadata SC(s) 620, or

make any necessary corrections and release the product for packing into a Metadata SC(s) 620 or

add a comment specifying the corrective action to take and resubmit the product to the Manual Metadata Entry Process 704

In another embodiment, after the creation of the SC(s), there is another optional quality

assurance step where the Content 113 of the SC(s) can be opened and examined for completeness and accuracy, and, at that time, final approval can be given or denied for the product's release to the retail channel.

D. Content Processing Tools

The Content Processing Tools 155 is actually a collection of software tools which are used to process the digital content file to create watermarked, encoded, and encrypted copies of the content. The tools makes use of industry standard digital content processing tools to allow pluggable replacement of watermarking, encoding and encryption technologies as they evolve. If the selected industry tool can be loaded via a command line system call interface and passed parameters or provides a toolkit wherein functions can be called via a DLL interface, the content processing can be automated to some degree. A front end application to each tool queries the appropriate queue in the Content Processing Tools 155 for the next available job, retrieves the required files and parameters and then loads the industry standard content processing tool to perform the required function. Upon completion of the task, manual update to the queue may be required if the tool does not report terminating status.

A generic version of the Content Processing Tools 155 is described, but customization is possible. The Content Processing Tools 155 can be written in Java, C/C++ or any equivalent software. The Content Processing Tools 155 can be delivered by any computer readable means including diskettes, CDS or via a Web site.

1. Watermarking Tool

The Watermarking Tool provides a user the ability to implement the Watermarking Process 808 as described above. This tool applies copyright information of the Content 113 owner to the song file using audio Watermarking technology. The actual information to be written out is determined by the Content Provider(s) 101 and the specific watermarking technology selected. This information is available to the front end Watermarking Tool so that it can properly pass this information to the watermarking function. This imposes a synchronization requirement on the Metadata Assimilation and Entry Tool 161 to assure that it has acquired this information prior to,

for example, allowing the song's audio file to be processed. This song will not be available for audio processing until the watermarking information has been obtained.

The watermark is applied as the first step in audio processing since it is common to all encodings of the song created. As long as the watermark can survive the encoding technology, the watermarking process need only occur once per song.

Various watermarking technologies are known and commercially available. The front end Watermarking Tool though is capable of supporting a variety of industry Watermarking Tools.

2. Preprocessing and Compression Tool

The Preprocessing and Compression Tool provides a user the ability to implement the Preprocessing and Compression Process 809 as described above. Audio encoding involves two processes. Encoding is basically the application of a lossy compression algorithm against, for a music content example, a PCM audio stream. The encoder can usually be tuned to generate various playback bit stream rates based on the level of audio quality required. Higher quality results in larger file sizes and since the file sizes can become quite large for high quality Content 113, download times for high quality Content 113 can become lengthy and sometimes prohibitive on standard 28,800 bps modems.

The Content Provider(s) 101 may, therefore, choose to offer a variety of digital content qualities for download to appease both the impatient and low bandwidth customers who don t want to wait hours for a download and the audiophile or high bandwidth customers who either only buys high quality Content 113 or has a higher speed connection.

Compression algorithms vary in their techniques to generate lower bit rate reproductions of Content 113. The techniques vary both by algorithm (i.e. MPEG, AC3, ATRAC) and by levels of compression. To achieve higher levels of compression, typically the data is re-sampled at lower sampling rates prior to being delivered to the compression algorithm. To allow for more efficient compression with less loss of fidelity or to prevent drastic dropout of some frequency ranges, the digital content may sometimes require adjustments to equalization levels of certain frequencies or adjustments to the dynamics of the recording. The content preprocessing requirements are directly related to the compression algorithm and the level of compression required. In some cases, the style

of Content 113 (e.g. musical genre) can be successfully used as a base for determining preprocessing requirements since songs from the same genre typically have similar dynamics. With some compression tools, these preprocessing functions are part of the encoding process. With others, the desired preprocessing is performed prior to the compression.

Besides the downloadable audio file for sale, each song also has a Low Bit Rate (LBR) encoded clip to allow the song to be sampled via a LBR streaming protocol. This LBR encoding is also the responsibility of the Content Processing Tools 155. This clip is either provided by the Content Provider(s) 101 as a separate PCM file or as parameters of offset and length.

As with watermarking, it is hoped that the encoding tools can be loaded via a DLL or command line system call interface and passed all the required parameters for preprocessing and compression. The front end Encoding Tool may have a synchronization requirement with the Metadata Assimilation and Entry Tool 161, for example if the content is music, and if it is determined that the song's genre is acquired from the Database 160 of the Content Provider(s) prior to performing any audio preprocessing. This depends on the encoding tools selected and how indeterminate the genre for the song is. If the Content Provider(s) 101 varies the choice of encoded quality levels per song, this information is also be provided prior to the encoding step and agrees with the metadata being generated by the Metadata Assimilation and Entry Tool 161.

A variety of high quality encoding algorithms and tools are known today. The front end Encoding Tool though is capable of supporting a variety of industry encoding tools.

Turning now to FIG. 12 is shown a flow diagram of one embodiment for the Automatic Metadata Acquisition Tool of FIG. 8 according to the present invention. The process starts with reading an identifier from the media the Content Provider(s) 101 is examining. One example of content in an audio CD embodiment. In an audio CD embodiment, the following codes may be available Universal Price Code (UPC), International Standard Recording Code (ISRC), International Standard Music Number (ISMN). This identifier is read in the appropriate player for the content, for example an audio CD Player for audio CD, DVD player for DVD movie, DAT recorder for DAT recording and equivalent, step 1201. Next this Identifier is used to index a Database 160 for the Content Provider(s) 101, step 1202. Some or all of the information required by the Work Flow Manager Process as described in FIG. 8 is retrieved in Database 160 and any other related sources,

step 1203. This information can include the Content 113 and the metadata related to it. In step 1204, the additional information retrieved is used to start the Work Flow Manager 154 for creating electronic Content 113. It should be understood, that several selections of media, such as several audio CDS, can be queued up so as to enable the Automatic Metadata Acquisition Tool to create a series of Content 113 for electronic distribution. For example, all the Content 113 could be created from a series of CDS or even selected tracks from one or more CDS examined by the Content Provider(s) 101.

In an alternate embodiment, the preprocessing parameters can be retrieve from the Database 160 of the Content Provider(s) automatically. Referring now to FIG. 13 is a flow diagram of a method to automatically set the Preprocessing and Compression parameters of the Preprocessing and Compression Tool of FIG. 8 according to the present invention. In this embodiment. the Content 113 is music. In step 1301, music (Content 113) is selected to be encoded in Content Processing Tools 155. The genre of the music selected is determined, step 1302. This can be entered manually or by using other meta data available, such as the additional data retrieved from the process described in FIG. 12. The audio compression level and audio compression algorithms selected are than examined, step 1303. Next, a lookup is made by genre, compression settings and compression algorithms of what compression parameters should be used in the Preprocessing and Compression Process 809, 1304.

3. Content Quality Control Tool

The Content Quality Control Tool provides a user the ability to implement the Content Quality Control Process 810 as described above. This is an optional Content Processing Tool and provides an opportunity for a quality control technician to review the encoded and watermarked content files and approve or reject the content files based on quality judgments. He can re-encode the content making manual preprocessing adjustments until the quality is adequate or can flag the song for reprocessing and attach a note describing the problem.

This process step can be configured by the Content Provider(s) 101 as an optional or required step of the content processing work flow. An additional optional Final Quality Assurance Process 813 step is provided after packaging of all the SC(s) for this content (e.g. each SC(s) for songs on

a CD) at which time the quality of the content encoding can be tested but catching a problem early prior to encryption and packaging allows for more efficient content processing. It is, therefore, highly desirable that the content quality be assured at this step as opposed to waiting until final completion of all processing.

4. Encryption Tool

The Encryption Tool provides a user the ability to implement the Encryption Process 811 as described above. Content encryption is the final step of the Content Processing Tools 155. Each of the versions of the content that were created by the Encoding Tool is now encrypted. The encryption tool is a function of the SC(s) Packer. The SC(s) Packer is called to encrypt the song and returns the generated encryption key used. This key is later passed into the SC(s) Packer for use in creation of the Metadata SC(s) 620.

E. Content SC(s) Creation Tool

Once all metadata has been gathered the Content SC(s) Creation Tool groups the metadata into categories based on their intended use. These groups of metadata are written into files to be passed in to the SC(s) Packer Tool as Metadata parts for the Metadata SC(s) 620. Each part (file) has unique processing requirements. Once the associated songs have been processed and encrypted and the target destination (URL of Content Hosting Site(s) 111) has been determined, the Content SC(s) 630 for the Content 113 are ready to be created. The Content 113 which have completed processing and have met all the requirements described above, are queued for packing in the packer queue of the Work Flow Manager 154.

The Content SC(s) Creation Tool now retrieves all the required files created by the previous steps of the Metadata Assimilation and Entry Tool 161 and calls the SC(s) Packer functions to create the Metadata SC(s) 620 and Content SC(s) 630. This process creates a single Metadata SC(s) 620 and multiple Content SC(s) 630 for each song. For example, if the content is music, each of the audio files created during audio processing for the various quality levels of the full song is packed into separate Content SC(s) 630. The audio file created for the sample clip is passed as a metadata file to be included in the Metadata SC(s) 620.

F. Final Quality Assurance Tool

The Final Quality Assurance Tool provides a user the ability to implement the Final Quality Assurance Process 813 as described above. Once all the SC(s) have been built for a content file, the content is available for a final quality assurance check. Quality assurance can be performed at various stages of the Content 113 preparation process. The Content Provider(s) 101 can choose to perform quality assurance as each major step is completed to prevent excessive rework later or may choose to wait until all audio preparation processes are complete and perform quality assurance on everything at once. If the latter is chosen, quality assurance is performed at this point upon completion of the creation of the SC(s). This tool allows each SC(s) for the song to be opened, examined, and the audio played.

Any problem discovered, even minor text changes requires that the SC(s) be rebuilt due to internal security features of SC(s). To avoid unnecessary re-processing time, it is highly recommended that the interim quality assurance steps be utilized to assure accuracy of the metadata and that this specific quality assurance step be reserved for validating appropriate cross references between the SC(s) associated with this song. If problems are found, the assurer can enter a problem description to be attached to the song and have it re-queued to the appropriate processing queue for reprocessing. Status is updated appropriately in the Work Flow Manager 154 to indicate the status of all related components of the song. If no problems are discovered, the Content 113 is marked or flagged as ready for release.

G. Content Dispersement Tool

The Content Dispersement Tool provides a user the ability to implement the Content Dispersement Process 814 as described above. Once the Content 113 has been approved for release, the SC(s) for the Content 113 are placed in the queue of the Content Dispersement Process. The Content Dispersement Tool monitors the queue and performs immediate transfer of the SC(s) files or batch transfer of a group of SC(s) files based on the configuration settings provided by the Content Provider(s) 101. The Content Provider(s) 101 can also optionally configure the Content Dispersement Tool to automatically hold all SC(s) in this queue until they are manually flagged for release. This allows the Content Provider(s) 101 to prepare content in advance of their scheduled

release date and hold them until they wish to release them e.g., a new song, movie or game. The SC(s) can also control access to Content 113 based on a defined release date so there is no requirement for the Content Provider(s) 101 to actually hold up delivery of the SC(s) but this manual release option can still be used for this purpose or used to manage network bandwidth required to transfer these large files.

When flagged for release, the Content SC(s) 630 for the Content 113 are transferred via FTP to the designated Content Hosting Site(s) 111. The Metadata SC(s) 620 is transferred via FTP to the Content Promotions Web Site 156. Here the SC(s) are staged to a new Content 113 directory until they can be processed and integrated into the Content Promotions Web Site 156.

FIG. 17 is a flow diagram of an alternate embodiment to automatically retrieve additional information for the Automatic Metadata Acquisition Tool of FIG. 8 according to the present invention. The process is similar for that described in FIG. 8 above. However, the quality checks of Supervised Release 806 and Content Quality Control 809 are combined into one quality check called Quality Control 1704. Performing quality checks prior to Metadata SC Creation 807 and Content SC Creation 812. Performing quality check prior to SC creation, eliminates the steps of unpacking the Content 113 and the associated Metadata SC(s) 620. In addition, in this embodiment, the queue of Products Awaiting Action/Information 801 have been eliminated. The jobs are placed on the specific process queues depending on what action is being requested. For example, if the job requires Manual Metadata, i.e. additional Metadata to be entered, the job is place on the Manual Metadata entry queue. Also the Automatic Metadata Acquisition 803 has been merged with New Content Request to occur up front prior to the Metadata Assimilation and Entry Tool 161 and the Content Processing Tool 155. Finally, it is important to point out that the Usage Conditions 804 are entered both at the Automatic Metadata Acquisition 803 and during the Manual Metadata Entry 803. Since, many of the usage conditions can be automatically filled-in during the Automatic Metadata Acquisition 803 step.

H. Content Promotions Web Site

To most effectively disperse information on what the Content Provider(s) 101 is making available for sale via digital download, and to get the necessary files to the Electronic Digital Content Store(s) 103 to enable it to make this Content 113 available for download to its customers, each

Content Provider(s) 101 should have a secure web site housing this information. This is similar to the method used today by some Content Provider(s) 101 to make promotional content available to their retailers and others with a need for this information. In the case where this type of service already exists, an additional section can be added to the web site where Electronic Digital Content Store(s) 103 can go to see a list of the content available for sale via download.

The Content Provider(s) 101 has complete control over the design and layout of this site or can choose to use a turnkey web server solution provided as part of the toolkit for Secure Digital Content Electronic Distribution System 100. To implement their own design for this service, the Content Provider(s) 101 need only provide links to the Metadata SC(s) 620 for Electronic Digital Content Store(s) 103 who access their site. This is accomplished using the toolkit for the Secure Digital Content Electronic Distribution System 100. The selection process and what information is shown is the discretion of the Content Provider(s) 101.

Metadata SC(s) 620 received into a new content directory via FTP from the Content Dispersement Tool is processed by the Content Promotions Web Site 156. These containers can be opened with the SC(s) Preview Tool to display or extract information from the container. This information can then be used to update HTML Web pages and/or add information to a searchable database maintained by this service. The SC(s) Preview Tool is actually a subset of the Content Acquisition Tool used by the Electronic Digital Content Store(s) 103 to open and process Metadata SC(s) 620. See the Content Acquisition Tool section for more details. The Metadata SC(s) 620 file should then be moved to a permanent directory maintained by the Content Promotions Web Site 156.

Once the Metadata SC(s) 620 has been integrated into the Content Promotions Web Site 156, its availability is publicized. The Content Provider(s) 101 can send a notification to all subscribing Electronic Digital Content Store(s) 103 as each new Metadata SC(s) 620 is added to the site or can perform a single notification daily (or any defined periodicity) of all Metadata SC(s) 620 added that day (or period). This notification is performed via a standard HTTP exchange with the Electronic Digital Content Store(s) 103 Web Server by sending a defined CGI string containing parameters referencing the Metadata SC(s) 620 added. This message is handled by the Notification Interface Module of the Electronic Digital Content Store(s) 103 which is described later.

I. Content Hosting

The Entertainment Industry produces thousands of content titles, such as CDS, movies and games every year, adding to the tens of thousands of content titles that are currently available. The Secure Digital Content Electronic Distribution System 100 is designed to support all of the content titles available in stores today.

The numbers of content titles that the Secure Digital Content Electronic Distribution System 100 may eventually download to customers on a daily basis is in the thousands or tens of thousands. For a large number of titles, this requires a large amount of bandwidth. The computer disk space and bandwidth needs call for a distributed, scalable implementation with multiple Content Hosting Site(s) 111. The system also supports customers all over the world. This requires overseas sites to speed delivery to the global customers.

Content hosting on the Secure Digital Content Electronic Distribution System 100 is designed to allow the Content Provider(s) 101 to either host their own Content 113 or share a common facility or a set of facilities.

Content hosting on the Secure Digital Content Electronic Distribution System 100 consists of multiple Content Hosting Site(s) 111 that collectively contain all of the Content 113 offered by the Secure Digital Content Electronic Distribution System 100 and several Secondary Content Sites (not shown) that contain the current hot hits offered by the Content Provider(s) 101. The number of Content Hosting Site(s) 111 changes depending on the number of End-User(s) using the system. The Secondary Content sites host a limited number of songs, but they will represent a large percentage of the bandwidth used on the system. The secondary sites are brought on line as the volume on the primary sites increases to the point of maximum capacity. The secondary sites can be located close to Network Access Points (NAPs) which helps speed up download times. They may also be placed in different geographic areas around the world to speed up download times.

Should the Content Provider(s) 101 choose to host all of their Content 113 in their own system, they can act as a single Content Hosting Site 111 with or without additional Secondary Content Sites. This allows them to build their own scalable distributed system. In another embodiment, Electronic Digital Content Store(s) 103 can also act as Content Hosting Site(s) 111 for certain Content 113. This embodiment requires a special financial agreement between the Electronic Digital Content

Store(s) 103 and the Content Provider(s) 101.

1. Content Hosting Sites

Content 113 is added to the Content Hosting Site(s) 111 via FTP or HTTP by the Content Disbursement Tool described in the Content Provider(s) Section of this specification or via offline means such as content delivery on tape, CD Rom, flash, or other computer readable media. The Metadata SC(s) 620 created by the Content Provider(s) 101 contain a field that indicates the URL locating the Content SC(s) 630 for this Content 113. This URL corresponds to a Content Hosting Site(s) 111. Electronic Digital Content Store(s) 103 can override this URL if allowed by the Content Provider(s) 101 in the Offer SC(s) 641. The End-User Device(s) 109 communicates to this Content Hosting Site(s) 111 when it wants to download the Content SC(s) 630.

The End-User Device(s) 109 initiates the request for a Content SC(s) 630 by sending the License SC(s) 660 to the Content Hosting Site(s) 111. This is the same License SC(s) 660 returned by the Clearinghouse(s) 105. The Digital Signature of the License SC(s) 660 can be verified to determine if it is a valid License SC(s) 660. If it is a valid License SC(s) 660 either the download is initiated, or the download request may be redirected to another Content Hosting Site(s) 111.

2. Content Hosting Site(s) 111 provided by the Secure Digital Content Electronic Distribution System 100

For the Secure Digital Content Electronic Distribution System 100 the decision of which site should be used to download the Content 113 is made by the primary content site that received the initial request for a Content SC(s) 630. This site uses the following information to make this decision:

- Are there secondary content sites that host the Content 113 requested? (The majority of Content 113 offered by the Secure Digital Content Electronic Distribution System 100 is only located at primary sites);
- Where is the End-User Device(s) 109 geographically located? (This information can be obtained from the End-User Device(s) 109 when the request is initiated at the End-User Device(s) 109, this is passed up to the Clearinghouse(s) 105 in the Order SC(s) 650;

- Is the appropriate secondary site up and operational? (Sometimes the secondary sites may be off-line);
- What is the load of the secondary sites? (In some cases where a secondary site is swamped with activity another site that is less busy may be selected.

Before transmitting the Content SC(s) 630 to the End-User Device(s) 109, analysis and verifications are performed on the End-User's request. A database is kept of all of the License SC IDs that have been used to download Content 113. This database can be checked to ensure that the End-User Device(s) 109 only makes one request for each piece of Content 113 purchased. This prevents malicious users from repeatedly accessing the Content Hosting Site(s) 111 in hopes of slowing down the Content Hosting Site(s) 111 and prevents unauthorized download of the Content SC(s) 630.

The promotion and demotion of Content 113 to the Secondary Content sites is done periodically based on customer demand for the individual pieces of Content 113.

Content Hosting Router

The Content Hosting Router (not shown) resides in the Content Hosting Site(s) 111 and receives all requests from End-User(s) wanting to download Content 113. It performs validation checks on the End-User(s) request to ensure they indeed bought the Content 113. A database is maintained on the status of the Secondary Content Sites that includes what Content 113 is on them and their current status. This current status includes the amount of activity on the sites and whether a site is down for maintenance.

The only interface to the Content Hosting Router is the License SC(s) 660 that is sent by the End-User Device(s) 109 when Content 113 is required to be downloaded. The License SC(s) 660 includes information that indicates the user is allowed to download the Content 113.

Secondary Content Sites

The Secondary Content Sites (not shown) host the popular Content 113 of the Secure Digital Content Distribution System 100. These sites are geographically dispersed across the world and are located near Network Access Points (NAPs) to improve download times. These sites are added to

the system as demand on the primary Content Hosting Site(s) 111 nears maximum capacity

IX. ELECTRONIC DIGITAL CONTENT STORE(S)

A. Overview - Support for Multiple Electronic Digital Content Store(s) 103

Electronic Digital Content Store(s) 103 are essentially the retailers. They are the entities who market the Content 113 to be distributed to the customer. For distribution of Content 113, this would include Digital Content Retailing Web Sites, Digital Content Retail Stores, or any business who wishes to get involved in marketing electronic Content 113 to consumers. These businesses can market the sale of electronic Content 113 only or can choose to just add the sale of electronic goods to whatever other merchandise they currently offer for sale. Introduction of downloadable electronic goods into the service offering of the Electronic Digital Content Store(s) 103 is accomplished via a set of tools developed for the Electronic Digital Content Store(s) 103 as part of the Secure Digital Content Electronic Distribution System 100.

These tools are used by the Electronic Digital Content Store(s) 103 to:

- acquire the Metadata SC(s) 620 packaged by the Content Provider(s) 101
- extract Content 113 from these SC(s) to be used as input to building their service offering
- create Offer SC(s) 641 describing the downloadable Content 113 they are offering for sale
- handle the acknowledgment of the sale and initiation of the download by creating and sending Transaction SC(s) 640 to the End-User Device(s) 109
- manage a transaction log of sales of downloadable Content 113 and the status of each download
- handle status notifications and transaction authentication requests
- perform account reconciliation

The tools are designed to allow flexibility in how the Electronic Digital Content Store(s) 103 wishes to integrate sale of downloadable electronic Content 113 into its service. The tools can be used in such a way as to request that all financial settlements for downloadable Content 113 purchased be handled by the Clearinghouse(s) 105 although this is not required. These tools also enable Electronic Digital Content Store(s) 103 to completely service their customers and handle the

financial transactions themselves, including providing promotions and special offers. The tools enable the Electronic Digital Content Store(s) 103 to quickly integrate the sale of downloadable Content 113 into its existing services. In addition, the Electronic Digital Content Store(s) 103 is not required to host the downloadable Content 113 and does not have to manage its dispersement. This function is performed by the Content Hosting Site(s) 111 selected by the Content Provider(s) 101.

The tools for the Electronic Digital Content Stores(s) 103 are implemented in Java in the preferred embodiment but other programming languages such as C/C++, Assembler and equivalent can be used. It should be understood that the tools described below for the Electronic Digital Content Stores(s) 103 can run on a variety of hardware and software platforms. The Electronic Digital Content Stores(s) 103 as a complete system or as any of it's constitute components may be distributed as an application program in a computer readable medium including but not limited to electronic distribution such as the web or on floppy diskettes, CD ROMS and removable hard disk drives.

In another embodiment, the components of the Electronic Digital Content Stores(s) 103 is part of a programmer's software toolkit. This toolkit enables predefined interfaces to the components of the generic Electronic Digital Content Stores(s) 103 components and tools discussed below. These predefined interfaces are in the form of APIs or Application Programming Interfaces. A developer using these APIs can implement any of the functionality of the components from a high level application program. By providing APIs to these components, a programmer can quickly develop a customized Electronic Digital Content Stores(s) 103 without the need to re-created these functions and resources of any of these components.

Electronic Digital Content Store(s) 103 are not limited to Web based service offerings. The tools provided are used by all Electronic Digital Content Store(s) 103 wishing to sell downloadable electronic Content 113 regardless of the transmission infrastructure or delivery mode used to deliver this Content 113 to End-User(s). Broadcast services offered over satellite and cable infrastructures also use these same tools to acquire, package, and track electronic Content 113 sales. The presentation of electronic merchandise for sale and the method in which these offers are delivered to the End-User(s) is the main variant between the broadcast based service offering and the point-to-point interactive web service type offering.

B. Point-to-Point Electronic Digital Content Distribution Service

Point-to-Point primarily means a one-to-one interactive service between the Electronic Digital Content Store(s) 103 and the End-User Device(s) 109. This typically represents an Internet web based service provided via telephone or cable modem connection. Networks other than the Internet are supported in this model as well, as long as they conform to the Web Server/Client Browser model. FIG. 9 is a block diagram illustrating the major tools, components and processes of an Electronic Digital Content Store(s) 103.

1. Integration Requirements

The Secure Digital Content Electronic Distribution System 100 not only creates new online businesses but provides a method for existing businesses to integrate the sale of downloadable electronic Content 113 to their current inventory. The suite of tools provided to the Electronic Digital Content Store(s) 103 simplify this integration effort. The Content Acquisition Tool 171 and SC(s) Packer Tool 153 provides a method for the Electronic Digital Content Store(s) 103 to acquire information from the participating Content Provider(s) 101 on what they have available for sale and to create the files required to reference these downloadable objects as items in their own inventory. This process is batch driven and can be largely automated and is executed only to integrate new Content 113 into the site.

The tools for the Secure Digital Content Electronic Distribution have been designed to allow integration of sale of electronic downloadable Content 113 into typical implementations of web based Electronic Digital Content Store(s) 103 (i.e. Columbia House online, Music Boulevard, @Tower) and equivalent with minimal change to their current Content 113 retailing paradigm. Several methods of integration are possible and in the preferred embodiment, the Electronic Digital Content Store(s) 103 provides support for all product searches, previews, selections (shopping cart), and purchases. Each Electronic Digital Content Store(s) 103 establishes customer loyalty with its customers and continues to offer its own incentives and market its products as it does today. In the Secure Digital Content Electronic Distribution System 100, it would simply need to indicate which products in its inventory are also available for electronic download and allow its customers to select the electronic download option when making a purchase selection. In another embodiment, the

customer's shopping cart could contain a mixture of electronic (Content 113) and physical media selections. After the customer checks out, and the Electronic Digital Content Store(s) 103 has completed the financial settlement and logged or notified its shipping and handling functions to process the physical merchandise purchased, the commerce handling function of the Electronic Digital Content Store(s) 103 then calls the Transaction Processor Module 175 to handle all electronic downloads. It simply passes the required information and all processing from that point on is handled by the toolset for the Secure Digital Content Electronic Distribution System 100. In another embodiment, other methods of transaction handling are also possible using tools for the Secure Digital Content Electronic Distribution System 100 to handle the financial settlement should the Electronic Digital Content Store(s) 103 wish to sell downloadable merchandise only or to segregate the financial settlement of physical and downloadable merchandise.

To handle the downloading of merchandise, the Electronic Digital Content Store(s) 103 is given a Product ID (not shown) for each downloadable product that it acquires from the Content Promotions Web Site 156 for the Content Provider(s) 101. This Product ID is associated to a customer's purchase selection to the downloadable product. The Product ID is what the Electronic Digital Content Store(s) 103 passes to the Transaction Processor Module 175 to identify the product that the user has purchased. The SC(s) (Offer SC(s) 641) that were created to describe the products, are isolated from the Electronic Digital Content Store(s) 103 and kept in an Offer Database 181 in an effort to simplify management of these objects and make their existence transparent to the Electronic Digital Content Store(s) 103.

The Transaction Processor Module 175 and other additional functions are provided as web server side executables (i.e. CGI and NSAPI, ISAPI callable functions) or simply APIs into a DLL or C object library. These functions handle run time processing for End-User(s) interactions and optional interactions with the Clearinghouse(s) 105. These functions interact with the web server's commerce services to create and download to the End-User Device(s) 109 the files necessary to initiate the Content 113 download process. They also handle optional interactions to provide authorizations and accept notifications of completion of activities.

An Accounting Reconciliation Tool 179 is also provided to assist the Electronic Digital Content Store(s) 103 in contacting the Clearinghouse(s) 105 to reconcile accounts based on its own

and the transaction logs of the Clearinghouse(s) 105.

2. Content Acquisition Tool 171

The Content Acquisition Tool 171 is responsible for interfacing with the Content Promotions Web Site 156 to preview and download Metadata SC(s) 620. Since the Content Promotions site is a standard web site, a web browser is used by the Electronic Digital Content Store(s) 103 to navigate this site. The navigation features varies based on the site design of the Content Provider(s) 101. Some sites may provide extensive search capabilities with many screens of promotional information. Others may have a simple browser interface with lists of titles, performers or new releases to select from. All sites include the selection of Metadata SC(s) 620 containing all the promotional and descriptive information of a song or album.

Alternatively, the Electronic Store(s) 103 may subscribe to content updates and receive updates automatically via FTP.

Viewing Metadata

The Content Acquisition Tool 171 is a web browser helper application which launches whenever a Metadata SC(s) 620 link is selected at the Content Promotions Web Site 156. Selection of the SC(s) causes it to be downloaded to the Electronic Digital Content Store(s) 103, and launch the helper application. The Content Acquisition Tool 171 opens the Metadata SC(s) 620 and display the non-encrypted information contained therein. Displayed information includes Extracted Metadata 173, for a music example, the graphic image(s) associated with the song and the information describing the song, a preview clip of the song can also be listened to if included in the Metadata SC(s) 620. In an example where the Content 113 is music, promotional information about the song or album, the album title, and the artist is also shown if provided by the Content Provider(s) 101. This information is displayed as a series of linked HTML pages in the browser window. Purchasable Content 113 such as the song and the lyrics and whatever other metadata the Content Provider(s) 101 wishes to protect, is not accessible to the Retail Content Web Site 180.

In another embodiment, the Content Provider(s) 101 provides optional promotional content for a fee. In this embodiment such promotional content is encrypted in the Metadata SC(s) 620.

Financial settlement to open this data can be handled via the Clearinghouse(s) 105 with the account for the Electronic Digital Content Store(s) 103 being charged the designated fee.

Extracting Metadata

Besides the preview capabilities, this tool provides two additional features: metadata extraction and preparation of an Offer SC(s) 641. Selection of the metadata extraction option prompts the Electronic Digital Content Store(s) 103 to enter the path and filenames to where the metadata is to be stored. Binary metadata such as graphics and the audio preview clip is stored as separate files. Text metadata is stored in an ASCII delimited text file which the Retail Content Web Site 180 can then import into its database. A table describing the layout of the ASCII delimited file is also be created in a separate TOC file. Additional options is available to allow extraction into other National Language Support (NLS) supported formats.

One important piece of information provided in the extracted data is the Product ID. This Product ID is what the commerce handling function for the Electronic Digital Content Store(s) 103 needs to identify to the Transaction Processor Module 175 (for more information refer to Transaction Processing section), the Content 113 that the user has purchased. The Transaction Processor Module 175 uses this Product ID to properly retrieve the appropriate Offer SC(s) 641 from the Offer Database 181 for subsequent download to the End-User Device(s) 109. The Electronic Digital Content Store(s) 103 has full control over how it presents the offer of downloadable Content 113 on its site. It only needs to retain a cross reference of the Content 113 being offered to this Product ID to properly interface with the tools for the Secure Digital Content Electronic Distribution System 100. Providing this information here, allows the Electronic Digital Content Store(s) 103 to integrate this product or Content 113 into its inventory and sales pages (database) in parallel with the Offer SC(s) 641 creation process since both processes uses the same Product ID to reference the product. This is described further below.

Offer SC(s) Creation Packer 153

The Electronic Digital Content Store(s) 103 is required to create an Offer SC(s) 641 describing the downloadable Content 113 that is for sale. Most of the information that goes into the Offer SC(s)

641 is derived from the Metadata SC(s) 620. The Content Acquisition Tool 171 creates the Offer SC(s) 641 by:

- removing parts from the Metadata SC(s) 620 that are not required to be included in the Offer SC(s) 641 as defined by the Offer SC(s) Template in the Metadata SC(s) 620
- adding additional required parts as defined by defaults specified by the configuration options in this tool for the Electronic Digital Content Store(s) 103
- prompting for additional required inputs or selections as defined by the Offer SC(s) Template in the Metadata SC(s) 620
- calling the SC(s) Packer 153 to pack this information into the SC(s) format

Metadata to be displayed by the Player Application 195 (further described later) on the End-User Device(s) 109 is kept in the Metadata SC(s) 620. Other promotional metadata that was only used by the Electronic Digital Content Store(s) 103 as input to his web service database is removed from the Metadata SC(s) 620. Rights management information provided by the Content Provider(s) 101, such as watermarking instructions, encrypted Symmetric Keys 623, and Usage Conditions 517 defining the permitted uses of the object, are also retained.

This stripped down Metadata SC(s) 620 is then included in the Offer SC(s) 641. The Electronic Digital Content Store(s) 103 also attaches its own Usage Conditions called Store Usage Conditions 519 or purchase options to the Offer SC(s) 641. This can be accomplished interactively or automatically through a set of defaults. If configured to be processed interactively, the Electronic Digital Content Store(s) 103 is prompted with the set of permitted object Usage Conditions 517 as defined by the Content Provider(s) 101. He then selects the option(s) he wishes to offer to his customers. These now become the new Usage Conditions or Store Usage Conditions 519. To process automatically, the Electronic Digital Content Store(s) 103 configures a set of default purchase options to be offered for all Content 113. These default options are automatically checked against the permitted Usage Conditions 517 defined by the Content Provider(s) 101 and is set in the Offer SC(s) 641 if there are no discrepancies.

Once the Offer SC(s) 641 is created, it is stored in an Offer Database 181 and is indexed with the Product ID pre-assigned in the Metadata SC(s) 620. This Product ID is used later by the Electronic Digital Content Store(s) 103 to identify the downloadable Content 113 being purchased

by a customer when interfacing with the Offer Database 181 to retrieve the Offer SC(s) 641 for packaging and transmittal to the End-User(s). See the Transaction Processor Module 175 section for more details.

In another embodiment, the Electronic Digital Content Store(s) 103 hosts the Content SC(s) 641 at his site. This embodiment requires changes to the Offer SC(s) 641 such as the replacement of the URL of the Content Hosting Site(s) 111 with the URL of the Electronic Digital Content Store(s) 103.

3. Transaction Processing Module 175

Electronic Digital Content Store(s) 103 directs billing to Clearinghouse(s) 105. Alternatively, the Electronic Digital Content Store(s) 103 may request financial clearance direct from the Clearinghouse(s) 105. There are two basic modes for processing End-User(s) purchase requests for downloadable Content 113. If the Electronic Digital Content Store(s) 103 does not wish to handle the financial settlement of the purchase and has no special promotions or incentives governing the sale of the merchandise and does not use a shopping cart metaphor for batching the purchase requests, it may opt to provide links on its Content 113 download pages directly to the Offer SC(s) 641 files. These Offer SC(s) 641 would have to have been built with retail pricing information included in the metadata. Also included in the Offer SC(s) 641 is a special HTML offer page presenting the purchase options with terms and conditions of the sale. This page is built from a template created when the Offer SC(s) 641 was built. When the End-User(s) clicks on the direct link to the Offer SC(s) 641, the Offer SC(s) 641 is downloaded to the browser End-User Device(s) 109 launching a helper application which opens the container and present the offer page included in the Offer SC(s) 641. This page contains a form to collect customer information including credit card information and purchase option selection. The form then gets submitted directly to the Clearinghouse(s) 105 for financial settlement and processing. Optionally, this form may contain the fields needed to use the End-User(s)' credit information or industry standard local transaction handler.

An embodiment where the Electronic Digital Content Store(s) 103 handles billing is now described. The more typical mode of handling purchase requests is to allow the Electronic Digital

Content Store(s) 103 to process the financial settlement and then submit the download authorization to the End-User(s). This method allows the Electronic Digital Content Store(s) 103 to integrate sale of downloadable Content 113 with other merchandise offered for sale at his site, allows batch processing of purchase requests with only one consolidated charge to the customer (via a shopping cart metaphor) instead of individual charges for each download request, and allows the Electronic Digital Content Store(s) 103 to directly track his customers buying patterns and offer special promotions and club options. In this environment, the offer of downloadable Content 113 is included in his shopping pages which get added to a shopping cart when selected by the End-User(s) and get processed and financially settled as is done in the Electronic Digital Content Store(s)' 103 current shopping model. Once the financial settlement is completed, the commerce handling process of the Electronic Digital Content Store(s) 100 then calls the Transaction Processor Module 175 to complete the transaction.

Transaction Processor Module 175

The role of the Transaction Processor Module 175 is to put together the information needed by the End-User Device(s) 109 to initiate and process the download of the Content 113 purchased. This information is packaged into a Transaction SC(s) 640 which is sent back to the End-User Device(s) 109 by the Web Server as the response to the purchase submission. The Transaction Processor Module 175 requires three pieces of information from the commerce handling process of the Electronic Digital Content Store(s) 103: the Product IDs for the Content 113 purchased, Transaction Data 642, and an HTML page or CGI URL acknowledging the purchase settlement.

The Product ID is the value provided to the Electronic Digital Content Store(s) 103 in the Metadata SC(s) 620 associated to the Content 113 just sold. This Product ID is used to retrieve the associated Offer SC(s) 641 from the Offer Database 181.

The Transaction Data 642 is a structure of information provided by the transaction processing function of the Electronic Digital Content Store(s) 103 which is later used to correlate the Clearinghouse(s) 105 processing with the financial settlement transaction performed by the Electronic Digital Content Store(s) 103 and to provide user identity information to be included in the watermark of the Content 113 downloaded to the End-User Device(s) 109. When the

Clearinghouse(s) 105 receives a valid Order SC(s) 650, it logs a transaction indicating the Content 113 that was sold, which Electronic Digital Content Store(s) 103 sold it and the associated Transaction Data 642 including the End-User's Name and a Transaction ID 535. The Transaction ID 535 provides a reference to the financial settlement transaction. This information is later returned by the Clearinghouse(s) 105 to the Electronic Digital Content Store(s) 103 for use in reconciling its accounts with the billing statements received from the Content Provider(s) 101 (or his agent). The Clearinghouse Transaction Log 178 can be used by the Content Provider(s) 101 to determine what Content 113 of his has been sold and enables him to create a bill to each Electronic Digital Content Store(s) 103 for royalties owed him. Other electronic means besides billing can alternatively be used to settle accounts between the Content Provider(s) 101 and Electronic Digital Content Store(s) 103.

The information provided in the Transaction SC(s) 640 and the security and integrity of the Transaction SC(s) 640 provide sufficient authenticity to the Clearinghouse(s) 105 that the purchase transaction is valid and thus no further validation is required prior to the logging of this sale by the Clearinghouse(s) 105. The Electronic Digital Content Store(s) 103, however, has the option to request authentication before its accounts are charged (transaction logged at the Clearinghouse(s) 105 indicating to the Content Provider(s) 101 that this Electronic Digital Content Store(s) 103 has collected money for the sale of this Content 113). This request for authentication/notification is indicated by a flag in the Transaction Data 642. In this scenario, the Clearinghouse(s) 105 contacts the Electronic Digital Content Store(s) 103 and receive authorization from the Electronic Digital Content Store(s) 103 before the charge to his account and the release of the encryption Key 623. The Transaction ID 535 is passed to the Electronic Digital Content Store(s) 103 from the Clearinghouse(s) 105 as part of this authentication request to enable the Electronic Digital Content Store(s) 103 to associate this request to a prior transaction performed with the End-User(s). This Transaction ID 535 can be any unique value the Electronic Digital Content Store(s) 103 wishes to use and is solely for its benefit.

The Transaction Data 642 also contains a customer name. This name can be from the user name field of the purchase form filled out by the user when making his purchase, or from information logged previously during some user registration process with the Electronic Digital Content Store(s) 103, or the official name obtained from credit card information associated with the

card used in this transaction. This name is later included in the License Watermark 527.

The Transaction Data 642 also contains the Store Usage Conditions 519 purchased by the End-User(s). This information is included in the License Watermark 527 and used by the End-User Device(s) 109 in Copy and Play Control.

The final parameter required by the Transaction Processor Module 175 is the HTML page or CGI URL acknowledging the purchase settlement. The purpose of this is to allow the Electronic Digital Content Store(s) 103 to respond to the End-User(s) with an acknowledgment of the financial settlement and whatever other information he wishes to include in the response. This HTML page or CGI URL is included in the Transaction SC(s) 640 and is displayed in the browser window of the End-User Device(s) 109 when the Transaction SC(s) 640 is received and processed.

The Transaction SC(s) 640 is the HTTP response to the End-User(s) from the Electronic Digital Content Store(s) 103 after processing the purchase submission. Sending a SC(s) as the direct HTTP response forces the automatic loading on the End-User Device(s) 109 of a SC(s) Processor Helper Application thus allowing automatic completion of the transaction without depending on further End-User(s) initiated actions. This process is described in more detail in the End-User Device(s) 109 and Player Application 195 section later.

When the Transaction Processor Module 175 is called with the required parameters, it builds a Transaction SC(s) 640 containing the Transaction Data 642, the transaction acknowledgment HTML page or reference URL other required security features of the SC(s), and retrieves and imbeds the Offer SC(s) 641 associated with the purchase. It also logs information about this transaction for later use by the Notification Interface Module 176 and the Account Reconciliation Tool 179.

4. Notification Interface Module 176

The Notification Interface Module 176 is a Web Server side executable routine (CGI or function callable by NSAPI, ISAPI or equivalent). It handles optional requests and notifications from the Clearinghouse(s) 105, the End-User Device(s) 109, the Content Hosting Site(s) 111, and the Content Provider(s) 101. The events that the Electronic Digital Content Store(s) 103 can optionally request notification for are:

- Notification from the Clearinghouse(s) 105 that the End-User Device(s) 109 requested an

encryption Key 623 and the Clearinghouse(s) 105 is releasing the encryption Key 623 for the specified Content 113. This notification can optionally be configured to require authentication from the Electronic Digital Content Store(s) 103 prior to the encryption Key 623 being sent to the End-User Device(s) 109.

- Notification from the Content Hosting Site(s) 111 that the Content SC(s) 630 has been sent to the End-User Device(s) 109.
- Notification from the End-User Device(s) 109 that the Content SC(s) 630 and the License SC(s) 660 have been received and successfully used to process the Content 113 or was found to be corrupt.
- Notification from the Content Provider(s) 101 that new Content 113 has been placed in the Content Promotions Web Site 156.

None of these notifications are a required step in the Secure Digital Content Electronic Distribution System flows 100 but are provided as options to allow the Electronic Digital Content Store(s) 103 the opportunity to close its records on the satisfaction of completion of the sale. It also provides information that may be needed to handle customer service requests by letting the Electronic Digital Content Store(s) 103 know what functions have transpired since financial settlement of the transaction or what errors occurred during an attempt to complete the sale. Alternatively, much of this status can be obtained from the Clearinghouse(s) 105 through the Customer Service Interface 184 as needed.

Frequency of notification of new Content 113 available at the Content Promotions Web Site 156 is determined by the Content Provider(s) 101. Notification may be provided as each new Metadata SC(s) 620 is added or just daily with all new Metadata SC(s) 620 added that day.

All of these notifications result in entries being made to the Transaction Log 178. If the Electronic Digital Content Store(s) 103 wishes to perform his own processing on these notifications, he can intercept the CGI call, perform his unique function and then optionally pass the request on to the Notification Interface Module 176.

5. Account Reconciliation Tool 179

This Account Reconciliation Tool 179 contacts the Clearinghouse(s) 105 to compare the

Transaction Log 178 with the log of the Clearinghouse(s) 105. This is an optional process which is available to help the Electronic Digital Content Store(s) 103 feel comfortable with the accounting for the Secure Digital Content Electronic Distribution System 100.

In another embodiment, this tool can be updated to provide electronic funds transfers for automated periodic payments to the Content Provider(s) 101 and the Clearinghouse(s) 105. It can also be designed to automatically process payments upon reception of an electronic bill from the Clearinghouse(s) 105 after reconciling the bill against the Transaction Log 178.

C. Broadcast Electronic Digital Content Distribution Service

Broadcast primarily refers to a one to many transmission method where there is no personal interaction between the End-User Device(s) 109 and the Electronic Digital Content Store(s) 103 to customize on-demand viewing and listening. This is typically provided over a digital satellite or cable infrastructure where the Content 113 is preprogrammed so that all End-User Device(s) 109 receive the same stream.

A hybrid model can also be defined such that an Electronic Digital Content Store(s) 103 provides a digital content service organized in such a way that it can offer both a web distribution interface via an Internet connection as well as a higher bandwidth satellite or cable distribution interface via a broadcast service, with a great deal of commonality to the site design. If the IRD backchannel serial interface were connected to the web, and the IRD supported web navigation, the End-User(s) could navigate the digital content service in the usual way via the backchannel Internet interface, previewing and selecting Content 113 to purchase. The user can select high quality downloadable Content 113, purchase these selections, and receive the required License SC(s) 660 all via an Internet connection and then request delivery of the Content 113 (Content SC(s) 630) over the higher bandwidth broadcast interface. The Web service can indicate which Content 113 would be available for download in this manner based on the broadcast schedule or could build the broadcast streams based totally on purchased Content 113. This method would allow a Web based digital content service to contract with a broadcast facility to deliver high quality Content 113 to users equipped with the proper equipment making a limited number of specific Content 113 (e.g. songs or CDS) available daily in this manner and the entire catalog available for download in lower

quality via the web interface.

Other broadcast models can be designed where there is no web interface to the End-User Device(s) 109. In this model, promotional content is packaged in specially formatted digital streams for broadcast delivery to the End-User Device(s) 109 (i.e. IRD) where special processing is performed to decode the streams and present the End-User(s) with the promotional content from which purchase selections can be made.

The actual purchase selections would still be initiated via backchannel communications from the End-User Device(s) 109 to the Clearinghouse(s) 105 and would utilize SC(s) to perform all data exchange. The toolset provided to the Electronic Digital Content Store(s) 103 has been architected and developed in such a way that most of the tools apply to both a point-to-point Internet service offering as well as a broadcast satellite or cable offering. The tools used by a Digital Content Web Site Electronic Digital Content Store(s) 103 to acquire and manage Content 113 as well as prepare SC(s) is also used by a satellite based Electronic Digital Content Store(s) 103 to manage and prepare Content 113 for distribution on a broadcast infrastructure. The SC(s) distributed over a Web service are the same as those distributed over a broadcast service.

X. END-USER DEVICE(S) 109

The applications in the End-User Device(s) 109 for the Secure Digital Content Electronic Distribution System 100 perform two main functions: first the SC(s) processing and copy control; and second playback of encrypted Content 113. Whether the End-User Device(s) 109 is a Personal Computer or a specialized electronic consumer device, it has to be capable of performing these base functions. The End-User Device(s) 109 also provides a variety of additional features and functions like creating play lists, managing the digital content library, displaying information and images during content playback, and recording to external media devices. These functions vary based on the services these applications are supporting and the type of devices the applications are designed for.

A. Overview

Referring now to FIG. 10, shown is the major components and processes and End-User

Device(s) 109 Functional Flow. The applications designed to support a PC based web interface Content 113 service consists of two executable software applications: the SC(s) Processor 192 and the Player Application 195. The SC(s) Processor 192 is an executable application which is configured as a Helper Application into the End-User(s) Web Browser 191 to handle SC(s) File/MIME Types. This application is launched by the Browser whenever SC(s) are received from the Electronic Digital Content Store(s) 103, the Clearinghouse(s) 105, and the Content Hosting Site(s) 111. It is responsible for performing all required processing of the SC(s) and eventually adding Content 113 to the Digital Content Library 196 of the End-User(s).

The Player Application 195 is a stand alone executable application which the End-User(s) loads to perform Content 113 in his Digital Content Library 196, manage his Digital Content Library 196 and create copies of the Content 113 if permitted. Both the Player Application 195 and SC(s) Processor 192 applications can be written in Java, C/C++ or any equivalent software. In the preferred embodiment, the applications can be downloaded from computer readable means such as website. However, other delivery mechanisms are also possible such as being delivered on computer readable media such as diskettes or CDS.

The searching and browsing of Content 113 information, previewing of, for example, song clips, and selecting songs for purchase is all handled via the End-User(s) Web Browser 191. Electronic Digital Content Store(s) 103 provides the shopping experience in the same way that is offered today by many Content 113 retailing web sites. The difference to the End-User(s) over today's web based Content 113 shopping is that they may now select downloadable Content 113 objects to be added to their shopping cart. If the Electronic Digital Content Store(s) 103 has other merchandise available for sale in addition to the downloadable objects, the End-User(s) may have a combination of physical and electronic downloadable merchandise in his shopping cart. The Secure Digital Content Electronic Distribution End-User Device(s) 109 are not involved until after the End-User(s) checks out and submits his final purchase authorization to the Electronic Digital Content Store(s) 103. Prior to this point, all interaction is between the Web Server for the Electronic Digital Content Store(s) 103 and the Browser 191 on the End-User Device(s) 109. This includes preview of sample Digital Content clips. Digital Content clips are not packaged into SC(s) but instead are integrated into the web service of the Electronic Digital Content Store(s) 103 as

downloadable files or fed from a streaming server. The format of the Content 113 clip is not dictated by the system architecture. In another embodiment, the Player Application 195 could interact directly with the Electronic Digital Content Store(s) 103 or Clearinghouse(s) 105 or offline using a promotional CD.

B. Application Installation

The Player Application 195 and the Helper Application 1981 are packaged into a self installing executable program which is available for download from many web sites. The Clearinghouse(s) 105 acts as a central location which hosts the master download page at a public web site. It contains links to the locations from which the installation package can be downloaded. The installation package is available at all Content Hosting Site(s) 111 to provide geographic dispersal of the download requests. Each participating Electronic Digital Content Store(s) 103 can also make the package available for download from their site or may just provide a link to the master download page at the public web site of the Clearinghouse(s) 105.

Any End-User(s) wishing to purchase downloadable Content 113, downloads and install this package. The installation is self contained in this downloadable package. It unpacks and installs both the Helper Application 198 and the Player Application 195 and also configure the Helper Application 198 to the installed Web Browser(s).

As part of the installation, a Public/Private Key 661 pair is created for the End-User Device(s) 109 for use in processing Order and License SC(s) 660. A random Symmetric Key (Secret User Key) is also generated for use in protecting song encryption keys in the License Database 197. The Secret User Key (not shown) is protected by breaking the key into multiple parts and storing pieces of the key in multiple locations throughout the End-User(s)' computer. This area of the code is protected with Tamper Resistant Software technology so as not to divulge how the key is segmented and where it is stored. Preventing access to this key by even the End-User(s) helps to prevent piracy or sharing of the Content 113 with other computers. See the SC(s) Processor 192 section for more details on how these keys are used.

Tamper-resistant software technology is a method to deter unauthorized entry into a computer software application by a hacker. Typically a hacker wants to understand and/or modify the software

to remove the restrictions on the usage. In practicality, no computer program exists that cannot be hacked; that is why tamper-resistant software is not called "tamper-proof". But the amount of effort required to hack a tamper-resistance protect application usually deters most hackers because the effort is not worth the possible gain. Here the effort would be to gain access to a key to one piece of Content 113, perhaps a single song on a CD.

One type of tamper-resistant software technology is from IBM. One product this code was introduced is in the IBM ThinkPad 770 laptop computer. Here, the tamper-resistant software was used to protect the DVD movie player in the computer. Digital Content Provider(s) such as Hollywood studios, concerned about the advent of digital movies and the ease at which perfect copies can be made, have insisted that movies on DVD disc(s) contain copy protection mechanisms. IBM's tamper-resistant software made it difficult to circumvent these copy protection mechanisms. This is a very typical application for tamper-resistant software; the software is used to enforce rules on the usage of some protected type of Content 113.

IBM's tamper-resistant software puts several types of obstacles in the path of the attacker. First, it contains techniques to defeat, or at least reduce the effectiveness of, the standard software tools that the hacker uses: debuggers and disassemblers. Second it contains self-integrity checking, so that single modifications, or even small handfuls of modifications, will be detected and cause incorrect operation. Finally, it contains obfuscations to mislead hackers regarding its true operation. The latter technique is largely ad hoc, but the first two build upon well-known tools in cryptography: encryption and digital signatures.

C. Secure Container Processor 192

When the End-User(s) submits the final purchase authorization to the Electronic Digital Content Store(s) 103 for the merchandise he has collected in his shopping cart, his Web Browser remains active waiting for a response from the Web Server. The Web Server at the Electronic Digital Content Store(s) 103 processes the purchase and performs the financial settlement and then returns a Transaction SC(s) 640 to the End-User Device(s) 109. The SC(s) Processor 192 (Helper Application 198) is launched by the Web Browser to process the SC(s) mime type associated with the Transaction SC(s) 640. FIG. 14 is an example of user interface screens of the Player

Application 195 downloading content to a local library as described in FIG. 10 according to the present invention.

The SC(s) Processor 192 opens the Transaction SC(s) 640 and extract the Response HTML page and Offer SC(s) 641 contained within. The Response HTML page is displayed in the Browser window acknowledging the End-User(s)' purchase. The Offer SC(s) 641 are then opened and the Content 113 (e.g. song or album) names along with the projected download times are extracted from them, step 1401. A new window is then displayed with this information and the End-User(s) is presented with options to schedule the download(s) of the Content 113 (e.g. for music, songs or entire albums), step 1402. The End-User(s) can select immediate download or can schedule the download to occur at a later time. If a later time is selected, the download schedule information is saved in a log and the download is initiated at the scheduled time if the End-User Device(s) 109 is powered on at that time. If the computer is not active at the scheduled download time or the communication link is not active, the End-User(s) is prompted to reschedule the download when the computer is next powered up.

When the scheduled download time occurs or if immediate download was requested, the SC(s) Processor 192 creates Order SC(s) 650 from information in the Transaction SC(s) 640, Offer SC(s) 641, and the Public Key 661 of the End-User(s) generated at install time. This Order SC(s) 650 is sent via HTTP request to the Clearinghouse(s) 105. When the Clearinghouse(s) 105 returns the License SC(s) 660, the Helper Application 198 is re-invoked to process the License SC(s) 660. The License SC(s) 660 is then opened and the URL of the Content Hosting Site(s) 111 is extracted from the referenced Order SC(s) 650. The License SC(s) 660 is then sent to the specified Content Hosting Site 111, via http request through the Browser, requesting download of the Content SC(s) 630. When the Content SC(s) 630 comes back to the Browser, the Helper Application 198 is re-invoked again. The SC(s) Processor 192 displays the name of the Content 113 being downloaded along with a download progress indicator and an estimated time to completion.

As the Content 113 is being received by the SC(s) Processor 192, it loads the Content 113 data into memory buffers for decryption. The size of the buffers depends on the requirements of the encryption algorithm and watermarking technology 193 and is the minimum size possible to reduce the amount of unencrypted Content 113 exposed to hacker code. As a buffer is filled, it is decrypted

using the Key 623 (corresponding to the Public Key 661) of the End-User(s) extracted from the License SC(s) 660, which itself is first decrypted using the Private Key. The decrypted buffer is then passed to the watermarking function.

The watermarking 193 extracts the watermarking instructions from the License SC(s) 660 and decrypt the instructions using the Private Key of the End-User(s). The watermarking data is then extracted from the License SC(s) 660 which includes transaction information such as the purchaser's name as registered with the Electronic Digital Content Store(s) 103 from which this Content 113 was purchased or derived from the credit card registration information if the Electronic Digital Content Store(s) 103 does not provide a registration function. Also included in the watermark is the purchase date and the Transaction ID 535 assigned by the Electronic Digital Content Store(s) 103 to reference the specific records logged for this transaction. The Store Usage Conditions 519 are also included to be used by the Copy Control of the Player Application 195.

The Watermarking 193 is protected with Tamper Resistant Code technology so as not to divulge the watermarking instructions thus preventing a hacker from discovering the location and technique of the watermark. This prevents removal or modification of the watermark by a hacker.

After inscribing any required watermark to this content buffer, the buffer is passed to the scrambling function for Re-Encryption 194. A processor efficient secure encryption algorithm such as IBM's SEAL encryption technology is used to re-encrypt the Content 113 using a random Symmetric Key. Once the download and Decryption and Re-Encryption 194 process is complete, the encryption Key 623 used by the Content Provider(s) 101 to originally encrypt the Content 113 is now destroyed and the new SEAL key is itself encrypted using the Secret User Key created and hidden at installation time. This new encrypted Seal Key is now stored in the License Database 107.

Unlike source performed at the Content Provider(s) 101 and user watermarking performed at the End User Device(s) 109 may need to become an industry standard to be effective. These standards are still evolving. The technology is available to allow control information to be embedded in the music and updated a number of times. Until such time as the copy control standards are more stable, alternative methods of copy control have been provided in the Secure Digital Content Electronic Distribution System 100 so that it does not rely on the copy control watermark in order to provide rights management in the consumer device. Storage and p lay/record

usage conditions security is implemented utilizing encrypted DC Library Collections 196 that are tied to the End User Device(s) 109 and protected via the Tamper Resistant Environment. Software hooks are in place to support copy control watermarking when standards have been adopted. Support exists today for watermarking AAC and other encoded audio streams at a variety of compression levels but this technology is still somewhat immature at this time to be put to use as a sole method of copy control.

The Decryption and Re-Encryption 194 process is another area of the code that is protected with Tamper Resistant Code technology so as not to divulge the original Content 113 encryption key, the new SEAL key, the Secret User Key, and where the Secret User Key segments are stored and how the key is segmented.

The process of Decryption and Re-Encryption 194 serves two purposes. Storing the Content 113 encrypted with an algorithm like SEAL enables faster than real-time decryption and requires much less processor utilization to perform the decryption than does a more industry standard type algorithm like DES. This enables the Player Application 195 to perform a real-time concurrent decryption-decode-playback of the Content 113 without the need to first decrypt the entire file for the Content 113 prior to decode and playback. The efficiency of the SEAL algorithm and a highly efficient decode algorithm, allows not only concurrent operation (streaming playback from the encrypted file) but also allows this process to occur on a much lower powered system processor. Thus this application can be supported on a End-User Device(s) 109 as low end as a 60MHz Pentium system and perhaps lower. Separating the encryption format in which the Content 113 is finally stored from the original encryption format, allows for greater flexibility in the selection of the original content encryption algorithm. Thus use of widely accepted and proven industry standard algorithms can be used thus further enhancing Digital Content Industry acceptance of the Secure Digital Content Electronic Distribution System 100.

The second purpose of this Decryption and Re-Encryption 194 process is to remove the requirement that the original master encryption Key 623, used by the Content Provider(s) 101 to encrypt this Content 113, be stored on every End-User Device(s) 109 which has licensed this Content 113. The encrypted master Key 623, as part of the License SC(s) 660, is only cached on the hard disk of the End-User Device(s) 109 for a very short time and is in the clear only in memory and for

a very short time. During this execution phase, the Key 623 is protected via Tamper Resistant Code technology. Not having to retain this Key 623 in any form on the End-User Device(s) 109 once this Decryption and Re-Encryption 194 phase has completed, greatly lessens the possibility of piracy from hackers.

Once the song has been re-encrypted, it is stored in the Digital Content Library 196. All metadata required for use by the Player Application 195, is extracted from the associated Offer SC(s) 641 and also stored in the Digital Content Library 196, step 1403. Any parts of the metadata which are encrypted, such as the song lyrics, are decrypted and re-encrypted in the same manner as described above for the other content. The same SEAL key used to encrypt the Content 113 is used for any associated metadata needing to be encrypted.

D. The Player Application 195

1. Overview

The Secure Digital Content Electronic Distribution Player Application 195 (referred to here as the Player Application 195) is analogous to both a CD, DVD or other Digital Content player and to a CD, DVD, or other digital content storage management system. At its simplest, it performs Content 113, such as playing songs or videos. At another level, it provides the End-User(s) a tool for managing his/her Digital Content Library 196. And just as importantly, it provides for editing and playing of collections of content, such as songs, (referred to here as Play-lists).

The Player Application 195 is assembled from a collection of components that may be individually selected and customized to the requirements of the Content Provider(s) 101 and Electronic Digital Content Store(s) 103. A generic version of the player is described, but customization is possible.

Referring now to FIG. 15 there is shown a block diagram of the major components and processes of the Player Application 195 running on End-User Device(s) 109 of FIG. 10.

There are several component-sets that make up the subsystems of the Player Object Manager 1501:

1. End-User Interface Components 1509
2. Copy/Play Management Components 1504

3. Decryption 1505, Decompression 1506, Playback Components 1507 and may include recording.

4. Data Management 1502 and Library Access Components 1503

5. Inter-application Communication Components 1508

6. Other miscellaneous (Installation, etc) Components

Components from within each of these sets may be selected, based on the requirements of:

- the platform (Windows, Unix, or equivalent)
- communications protocols (network, cable, etc)
- Content Provider(s) 101 or Electronic Digital Content Store(s) 103
- Hardware (CD, DVD, etc)
- Clearinghouse(s) 105 technology and more.

The sections below detail the various component sets. The final section details how these components are put together in the generic player, and discusses how the components can be customized.

In another embodiment, the components of the Player Application 195 and the SC(s) Processor 192 are available as part of a programmer's software toolkit. This toolkit enables predefined interfaces to the components of the generic player application listed above. These predefined interfaces are in the form of APIs or Application Programming Interfaces. A developer using these APIs can implement any of the functionality of the components from a high level application program. By providing APIs to these components, a programmer can quickly develop a customized Player Application 195 without the need to re-created these functions and resources of any of these components.

2. End-User Interface Components 1509

Components from this set combine to provide the on-screen manifestation of the Player Application 195. Note that the design establishes no definitive layout of these components. One such layout is provided in the generic player. Based on requirements from Content Provider(s) 101

and/or Electronic Digital Content Store(s) and other requirements, alternate layouts are possible.

This set is grouped into subgroups, starting with the components used to present End-User Display 1510 and handle controls called End-User Controls 1511 used for such low-level functions as audio playback , and presentation of metadata. Next, the End-User Display Component 1510 is further divided by special function groupings (Play-list, Digital Content Library), and then object-container components used for grouping and placing of those lower-level components.

Within the component listings below, any reference to creating CDS or copying of Content 113 to a CD or other recordable medium only applies to the case where the Player Application 195 has such functionality enabled. Also note that the term CD in that context is a generic one, that can also represent various other external recording devices, such as MiniDisc or DVD.

FIG. 16 is an example user interface screens of the Player Application 195 of FIG. 15 according to the present invention. Function for the End-User Controls 1511 include (corresponding screens of an End-User Interface are shown 1601-1605):

Controls for performing the Content 113:

- Play/Stop button
- Play button
- Stop button
- Pause button
- Skip forward button
- Skip backward button
- Volume control
- Track position control/display
- Audio channel volume level display and more.

Controls for the displaying metadata associated with the Content 113

- Cover Picture button
- Cover Picture object
- Artist Picture button
- Artist Picture object

- Track List button
- Track List Information object
- Track List Selector object (click to play)
- Track Name object
- Track Information object
- Track Lyrics button
- Track Lyrics object
- Track Artist Name object
- Track Credits button
- Track Credits object
- CD Name object
- CD Credits button
- CD Credits object
- Generic (Configurable) Metadata button
- Generic Metadata object and more.

Function for the End-User Display 1510 include (corresponding screens of an End-User Interface are shown 1601 - 1605):

Play-list of display container

- Play-list Management button
- Play-list Management window
- Digital Content search button
- Digital Content search Definition object
- Digital Content search Submit button
- Digital Content search Results object
- Copy Selected Search Result Item To Play-list button
- Play-list object (editable)
- Play-list Save button
- Play-list Play button

- Play-list Pause button
- Play-list Restart button
- Create CD from Play-list button and more.

Display of Digital Content Library 196

- Digital content library button
- Digital content librarian window
- Digital content categories button
- Digital content categories object
- By-artist button
- By-genre button
- By-label button
- By-category button
- Delete button
- Add-to-Play-list button
- Copy to CD button
- Song List object
- Song List display container and more

Containers and Misc.

- Player window container
- Audio controls container
- Metadata controls container
- Metadata display container
- Toolbar container object
- Sample button
- Download button
- Purchase button
- Record button

- Player Name object
- Label/Provider/Store Advertisement object
- Label/Provider/Store URL button
- Artist URL Button and more

3. Copy/Play Management Components 1504

These components handle set up of encryption keys, Watermark processing, Copy management, and more. Interfaces also exist for communication with the Clearinghouse(s) 105, transmission of purchase requests, and more, for special services such as pay per listen or cases where each access to the Content 113 is accounted for. Currently, the communications to the Clearinghouse(s) 105 functions are handled by the SC(s) Processor 192.

The use of the Content 113 by the Player Applications 195 on End User Device(s) 109 is logged into a database such as the License Database 197. The tracking of each use of Content 113 by the Player Application 195 can be transmitted to one or more logging sites such as the Clearing House(s) 105 or Content Provider(s) 101 or Electronic Digital Content Store(s) 103 or any site designated and coupled to Transmission Infrastructures 107. This transmission can be scheduled at predetermined times to upload the usage information to a logging site. One predetermined time contemplated is early in the morning when Transmission Infrastructures 107 may not be as congested with network traffic. The Player Application 195 using known techniques, wakes-up at a scheduled time, and transmit the information from the local logging database to the logging site. By reviewing the logging site information, the Content Provider(s) 101 can measure the popularity of their Content 113.

In another embodiment, the instead of logging the usage of Content 113 for later uploading to a logging site, the use of the Content 113 is uploaded to the logging site during every use of the Content 113. For example, when duplicating or copying the Content 113 stored at the End User Device(s) 109, on to an external device such as DVD Disc, digital tape, flash memory, mini Disc or equivalent read/writeable removable media, the use is updates to the logging site. This may be a precondition to copying the Content 113 in the usage conditions 206 that is transmitted when the Content 113 is purchased. This ensures the Content Provider(s) 101 can accurately track the usage

of their Content 113 during their playing, duplicating or other actions upon the Content 113.

In addition, other information about the Content 113 can be uploaded to the logging site. For example the last time (e.g., hour and day) the Content 113 was performed; how many times the Content 113 was performed; if the Content 113 has been duplicated or copied to an authorized external device such as DVD Disc, digital tape or mini-Disc. In cases where there are multiple distinct users of a single Player Application 195 on the End User Device(s) 109, such as different members of a family, the identifications of the user of the Content 113 is transmitted along with the usage information to the logging site. By reviewing the usage information uploaded to the logging site, the Content Provider(s) 101 can measure the popularity of the Content 113 base on the actual usage, the identification of the user and the number of times the Content 113 has been performed. The actual usage measurement makes this system more factual driven over systems using sampling methods, such as a Nielsen Rating scheme for televisions, or telephone surveys, where only a limited number of users are sampled at any one time and the results extrapolated. In this present embodiment, the actual usage can be measures for the users logging back onto a designated web site such as the Electronic Digital Content Store(s) 103 or Content Provider(s) 101.

4. Decryption 1505, Decompression 1506 and Playback Components 1506

These components use the keys acquired by the Copy/Play Management components to unlock the audio data acquired from the Data Management and Library Access components, apply the appropriate decompression to prepare it for playback, and use system audio services to play it. In an alternate embodiment, the audio data acquired from the Data Management and Library Access components may be copied to removable media such as CDS, diskettes, tapes or MiniDisks.

5. Data Management 1502 and Library Access Components 1503

These components are used to store and retrieve song data on various storage devices on the End-User(s)' system, as well as handle requests for information about the stored songs.

6. Inter-application Communication Components 1508

These components are used for coordination between the Secure Digital Content Electronic

Distribution Player and other applications (e.g., Browser, helper-app and/or plug-in, etc) that may invoke the Player Application 195, or that the Player Application 195 needs to use when carrying out its functions. For example, when a URL control is activated, it invokes the appropriate browser and instruct it to load the appropriate page.

7. Other Miscellaneous Components

Individual components that don t fall into the categories above (e.g., Installation) are grouped here.

8. The Generic Player

In this section the combining of the components above into a version of the Player Application 195 is discussed. This is just one of many different examples possible, since the Player Application 195 is designed for customization by being based on software objects. The Player Object Manager 1501 is a software framework holding all the other components together. As discussed in the sections above, the blocks below the Player Object Manager 1501 in this diagram are required for any player, but may be replaced by specialized versions depending on such things as form of encryption or scrambling being used, types of audio compression, access methods for the Content 113 library, and more.

Above the Player Object Manager 1501 are Variable Objects 1512, which are mostly derived from the metadata associated with the Content 113 being played or searched. These Variable Objects are made available to the End-User Device(s) 109 by way of the End-User Display 1510 and received input from the End-User Controls 1511. All objects are configurable, and the layouts of all containers are customizable. These objects may be implemented in C/C++, Java or any equivalent programming language.

Using the Player Application 195

The following embodiment is for an example where the Player Application 195 running on End-User Device(s) 109 is an audio player where Content 113 is music. It should be understood to those skilled in the art that other types of Content 113 can be supported by the Player Application

195. A typical audio enthusiast has a library of CDS holding songs. All of these are available within the Secure Digital Content Electronic Distribution System 100. The set of songs that have been purchased from Electronic Digital Content Store(s) 103 are stored within a Digital Content Library 196 on his or her system. The groupings of songs that are analogous to physical CDS are stored as Play-lists. In some cases a Play-list exactly emulates a CD (e.g., all tracks of a commercially available CD has been purchased from an Electronic Digital Content Store(s) 103 as an on-line version of the CD and is defined by a Play-list equivalent to that of the CD). But most Play-lists is put together by End-User(s) to group songs they have stored in the Digital Content Libraries on their systems. However for the purposes of the ensuing discussions, an example of a custom made music CD is used when the term a Play-list is mentioned.

When the End-User(s) starts the Player Application 195 explicitly, rather than having it start up via invocation from the SC(s) Processor 192 Application, it pre-loads to the last Play-list that was accessed. If no Play-lists exist in the Digital Content Library 196, the Play-list editor is started automatically (unless the user has turned off this feature via a preference setting). See The Play-list, below for further details.

The Player Application 195 may also be invoked with a specific song as an argument, in which case it immediately enters Song-play mode. Optionally, the song may be prepared for play but await action by the End-User(s) before proceeding. See Song Play, below for more on this situation.

The Play-list (corresponding screen of an End-User Interface 1603):

When the End-User(s) has invoked the Play-list function, these are the available functions:

- Open Play-list
- Digital Content Librarian is invoked to display a list of stored Play-lists for selection. Also see Digital Content Librarian below for more info.
- Edit Play-list
- Invokes the Play-list Editor (see below), primed with the current Play-list if one has been loaded already. Otherwise the editor creates an empty Play-list to start with.
- Run Play-list
- Songs are played one at a time starting with the selected song (or the beginning of the play-list,

if no song is selected). Options set in the Play-list Editor affect the sequencing of the playback. However there is controls available here to override those options for this play of the Play-list.

- Play song
- Only the selected song from the Play-list is played. See Song Play below for more info.
- Play-list Info
- Display information about the Play-list.
- Song Info
- Display information about the selected song within the Play-list.
- Visit web site
- Load web site associated with this Play-list into browser.
- Librarian
- Open the Digital Content Librarian window. Also see Digital Content Librarian below for more info.

The Play-list Editor (corresponding screen of an End-User Interface 1603):

When invoking the Play-list editor, these are the End-User(s)' options:

- View/Load/Delete Play-lists
- Digital Content Librarian is invoked to display a list of stored Play-lists for selection of one to load or delete. Also see Digital Content Librarian below for more info.
- Save Play-list
- Current version of Play-list is saved in the Digital Content Library 196.
- Delete Song
- Currently selected song is deleted from Play-list.
- Add Song
- Digital Content Librarian is invoked in song-search mode, for selection of song to add to the Play-list. Also see Digital Content Librarian below for more info.
- Set Song Information
- Display and allow changes to information about the selected song within the play-list. This information is stored within the Play-list, and does not alter information about the song stored within the Digital Content Library 196. These things can be changed:

- Displayed Song Title
- End-User(s) notes about the song
- Lead-in delay on playing the song
- Follow-on delay after playing the song
- Start-point within song when playing
- End-point within song when playing
- Weighting for random mode
- Volume adjustment for this song and more.

Set Play-list attributes: Display and allow changes to the attributes of this Play-list. These attributes may be set:

- Play-list title
- Play-list mode (random, sequential, etc)
- Repeat mode (play once, restart when done, etc)
- End-User(s) notes about this Play-list

Librarian (corresponding screen of an End-User Interface 1601):

- Open the Digital Content Librarian window. Also see Digital Content Librarian below for more info.

Song Play

When a song has been prepared for play, either by invoking the Player Application 195 with the song as an argument or by selecting a song for play from a Play-list or within the Digital Content Librarian, these are the End-User(s)' options: (corresponding screen of an End-User Interface 1601):

- Play
- Pause
- Stop
- Skip Backward
- Skip Forward
- Adjust Volume
- Adjust Track Position

- View Lyrics
- View Credits
- View CD Cover
- View Artist Picture
- View Track Information
- View other metadata
- Visit web site
- Play-list
- Librarian and more.

Digital Content Librarian

The Digital Content Librarian can be invoked implicitly when selecting songs or Play-lists (see above) or may be opened in its own window for management of the Song Library on the End-User(s)' system. In that case, these are the End-User(s)' options:

Working with songs:

Sort All by Artist, Category, Label, other

Select Songs by Artist, Category, Label, other

Add selected songs to Current Play-list

Copy Song to CD (if enabled)

Delete Song

Add Song to Category and more.

Work with Play-lists:

Sort by Name

Sort by Category

Search by Keyword

Search by Included Song Title

Load Selected Play-list

Rename Play-list

Delete Play-list

Create CD from Selected Play-list (if enabled) and more.

Although a specific embodiment of the invention has been disclosed, it will be understood by those having skill in the art that changes can be made to this specific embodiment without departing from the spirit and scope of the invention. The scope of the invention is not to be restricted, therefore, to the specific embodiment, and it is intended that the appended claims cover any and all such applications, modifications, and embodiments within the scope of the present invention.

What is claimed is:

The embodiments of the invention in which an exclusive property or privilege is claimed are defined as follows:

1. An electronic content management system comprising a clearinghouse that enables securely providing data, the clearinghouse being capable of communicating with a data system for securely providing data, the data system being capable of receiving both data encrypted with a first encryption key and an encrypted first encryption key, the encrypted first encryption key being the first encryption key encrypted by a second encryption key, the clearinghouse comprising:

decryption of a first decryption key from the encrypted first decrypting key; and

transfer of the decrypted first decrypting key to a data system.

2. The management system of claim 1 wherein the data system comprises:

means for encrypting data by a first encrypting key to generate encrypted data;

means for encrypting the first decrypting key by a second encrypting key to generate an encrypted first decrypting key;

means for transferring the encrypted data to a second system;

means for transferring the encrypted first decrypting key to a second system; and

means for transferring the encrypted first decrypting key to the clearinghouse that possesses a second decrypting key.

3. The management system of claim 2 further comprising, means for re-encrypting the first decrypting key by a third encrypting key prior to the transfer of the first decrypting key and wherein the means for transferring the first decrypting key to a second system is the means for transferring the decrypted and re-encrypted first decrypting key to a second system.

4. The management system of claim 3, wherein the third encrypting key is a public key of a second system.

5. The management system of claim 2, wherein the second encrypting key is a public key of the

clearinghouse and the second decrypting key is a corresponding private key of the clearinghouse.

6. The management system of claim 2, further comprising means for confirming that the data was paid for.

7. The management system of claim 2, further comprising means for authorizing transfer of the data prior to transferring the encrypted data decrypting key to a second system.

8. A method of securely providing data to a user's system, said method comprising the steps of:

encrypting the data using a first encrypting key;

encrypting a first decrypting key using a second encrypting key;

transferring the encrypted data, which has been encrypted with the first encrypting key, to the user's system;

transferring the encrypted first decrypting key, which has been encrypted with the second encrypting key, to the user's system;

transferring the encrypted first decrypting key, which has been encrypted with the second encrypting key, to a clearing house that possesses a second decrypting key;

decrypting the first decrypting key using the second decrypting key; and

transferring the decrypted first decrypting key to the user's system.

9. The method as defined in claim 8, wherein the step of transferring the decrypted first decrypting key includes the sub-steps of:

re-encrypting the first decrypting key using a third encrypting key;

transferring the decrypted and re-encrypted first decrypting key to the user's system; and

decrypting the re-encrypted first decrypting key using a third decrypting key.

10. The method as defined in claim 8, wherein the first encrypting key and the first decrypting key are symmetric keys.

11. The method as defined in claim 10, wherein the second encrypting key is a public key of the clearinghouse and the second decrypting key is a corresponding private key of the clearinghouse.

12. The method as defined in claim 11, wherein the step of transferring the decrypted first decrypting key includes the sub-steps of:

re-encrypting the first decrypting key using a third encrypting key, the third encrypting key being a public key of the user;

transferring the decrypted and re-encrypted first decrypting key to the user's system; and

decrypting the re-encrypted first decrypting key using a third decrypting key, the third decrypting key being a corresponding private key of the user.

13. The method as defined in claim 8, wherein the step of transferring the encrypted first decrypting key to the user's system is performed by an electronic merchant and includes the sub-steps of:

initiating a purchase of the data or a license for the data from the electronic merchant; and

sending the encrypted first decrypting key and purchase transaction data to the user's system.

14. The method as defined in claim 13, wherein the step of transferring the encrypted first decrypting key to the user's system further includes the sub-step of charging the user for the data or the license, and

the step of decrypting the first decrypting key is performed by the clearinghouse and includes the sub-steps of:

verifying that the user has paid for the data or the license; and

decrypting the first decrypting key using the second decrypting key.

15. The method as defined in claim 13, wherein the step of decrypting the first decrypting key is performed by the clearinghouse and includes the sub-steps of:

charging the user for the data or the license; and

decrypting the first decrypting key using the second decrypting key.

16. The method as defined in claim 8, further comprising the step of decrypting the data using the first decrypting key.

17. The method as defined in claim 8, wherein the data includes music data.

18. A method of securely providing data to a user's system, the data being encrypted so as to only be decryptable by a data decrypting key, the data decrypting key being encrypted using a first public key, and the encrypted data being accessible to the user's system, said method comprising the steps of:

transferring the encrypted data decrypting key to a clearing house that possesses a first private key, which corresponds to the first public key;

decrypting the data decrypting key using the first private key;

re-encrypting the data decrypting key using a second public key;

transferring the re-encrypted data decrypting key to the user's system, the user's system possessing a second private key, which corresponds to the second public key; and

decrypting the re-encrypted data decrypting key using the second private key.

19. The method as defined in claim 18, wherein the step of transferring the encrypted data decrypting key to a clearing house includes the sub-steps of:

transferring the encrypted data decrypting key to the user's system; and

subsequently transferring the encrypted data decrypting key from the user's system to the clearing house.

20. The method as defined in claim 19, wherein the sub-step of transferring the encrypted data decrypting key to the user's system is performed by an electronic merchant and includes the sub-steps of:

initiating a purchase of the data or a license for the data from the electronic merchant; and

sending the encrypted data decrypting key and purchase transaction data to the user's system.

21. The method as defined in claim 20, wherein the sub-step of transferring the encrypted data decrypting key to the user's system further includes the sub-step of charging the user for the data or the license, and

the step of decrypting the data decrypting key is performed by the clearinghouse and includes the sub-steps of:

verifying that the user has paid for the data or the license; and

decrypting the data decrypting key using the first private key.

22. The method as defined in claim 20, wherein the step of decrypting the data decrypting key is performed by the clearinghouse and includes the sub-steps of:

charging the user for the data or the license; and

decrypting the data decrypting key using the first private key.

23. The method as defined in claim 18, further comprising the step of decrypting the encrypted data using the data decrypting key.

24. A method of operating a clearinghouse to provide integrity in a channel of commerce that includes a provider, a distributor, and a purchaser, the provider producing data and encrypting the data so as to only be decryptable by a data decrypting key, the encrypted data being accessible to the purchaser, said method comprising the steps of:

encrypting the data decrypting key using a public key of the clearinghouse;

sending the encrypted data decrypting key from the provider to the distributor;

when the purchaser desires to purchase the data or a license to use the data, sending the encrypted data decrypting key from the distributor to the purchaser;

sending the encrypted data decrypting key from the purchaser to the clearing house;

decrypting the data decrypting key using a private key of the clearinghouse and re-encrypting the data decrypting key using a public key of the purchaser; and

sending the re-encrypted data decrypting key from the clearinghouse to the purchaser.

25. The method as defined in claim 24, wherein the distributor charges the user a fee before sending the encrypted data decrypting key to the purchaser, and

the clearinghouse verifies that the user has paid the fee before sending the re-encrypted data decrypting key to the purchaser.

26. The method as defined in claim 24, wherein the clearinghouse charges the user a fee before sending the re-encrypted data decrypting key to the purchaser.

27. The method as defined in claim 24, further comprising the steps of:

decrypting the re-encrypted data decrypting key using the private key of the purchaser; and

decrypting the encrypted data using the data decrypting key.

28. A system for securely providing data to a user's system, the system comprising:

a content system;

a first public key;

a first private key; which corresponds to the first public key;

a data encrypting key;

a data de-encrypting key for de-encrypting data encrypted using the data encrypting key;

first data encryption means for encrypting data so as to be decryptable only by a data decrypting key;

second data encryption means, using the first public key, for encrypting the decrypting key;

a clearing house;

first transferring means for transferring the data decrypting key which has been encrypted to the clearing house, wherein the clearinghouse possesses the first private key;

first decrypting means for decrypting the data decrypting key using the first private key;

a second public key;

a second private key; which corresponds to the second public key;

re-encryption means for re-encrypting the data decrypting key using the second public key;

second transferring means for transferring the re-encrypted data decrypting key to the user's

system, wherein the user's system possesses the second private key; and

second decrypting means for decrypting the re-encrypted data decrypting key using the second private key.

29. The system as defined in claim 28, wherein the first transferring means further comprises:

a third transfer means for transferring the encrypted data decrypting key to the user's system; and

a fourth transfer means for subsequently transferring the encrypted data decrypting key from the user's system to the clearinghouse.

30. The system as defined in claim 29, wherein the third transfer means is performed by an electronic merchant and further comprises:

initiating means for initiating a purchase of the data or a license for the data from the electronic merchant; and

sending means for sending the encrypted data decrypting key and purchase transaction data to the user's system.

31. The system as defined in claim 30, wherein the sending means further comprises:

charging means for charging the user for the data or the license; and

wherein the first decrypting means is performed by the clearinghouse and further comprises:

verifying means for verifying that the user has paid for the data or the license; and

third decrypting means for decrypting the data decrypting key using the first private key.

32. The system as defined in claim 30, wherein the first decrypting means is performed by the clearinghouse and further comprises:

charging means for charging the user for the data or the license; and

fourth decrypting means for decrypting the data decrypting key using the first private key.

33. The system in claim 28, further comprises:

33. The system in claim 28, further comprises:

data decrypting key decrypting means for decrypting the encrypted data using the data decrypting key.

Application number / numéro de demande: 2467998

Figures: 7, 14, 16.

Pages:

Unscannable items
received with this application
(Request original documents in File Prep. Section on the 10th floor)

Documents reçu avec cette demande ne pouvant être balayés
(Commander les documents originaux dans la section de préparation des dossiers au 10ème étage)

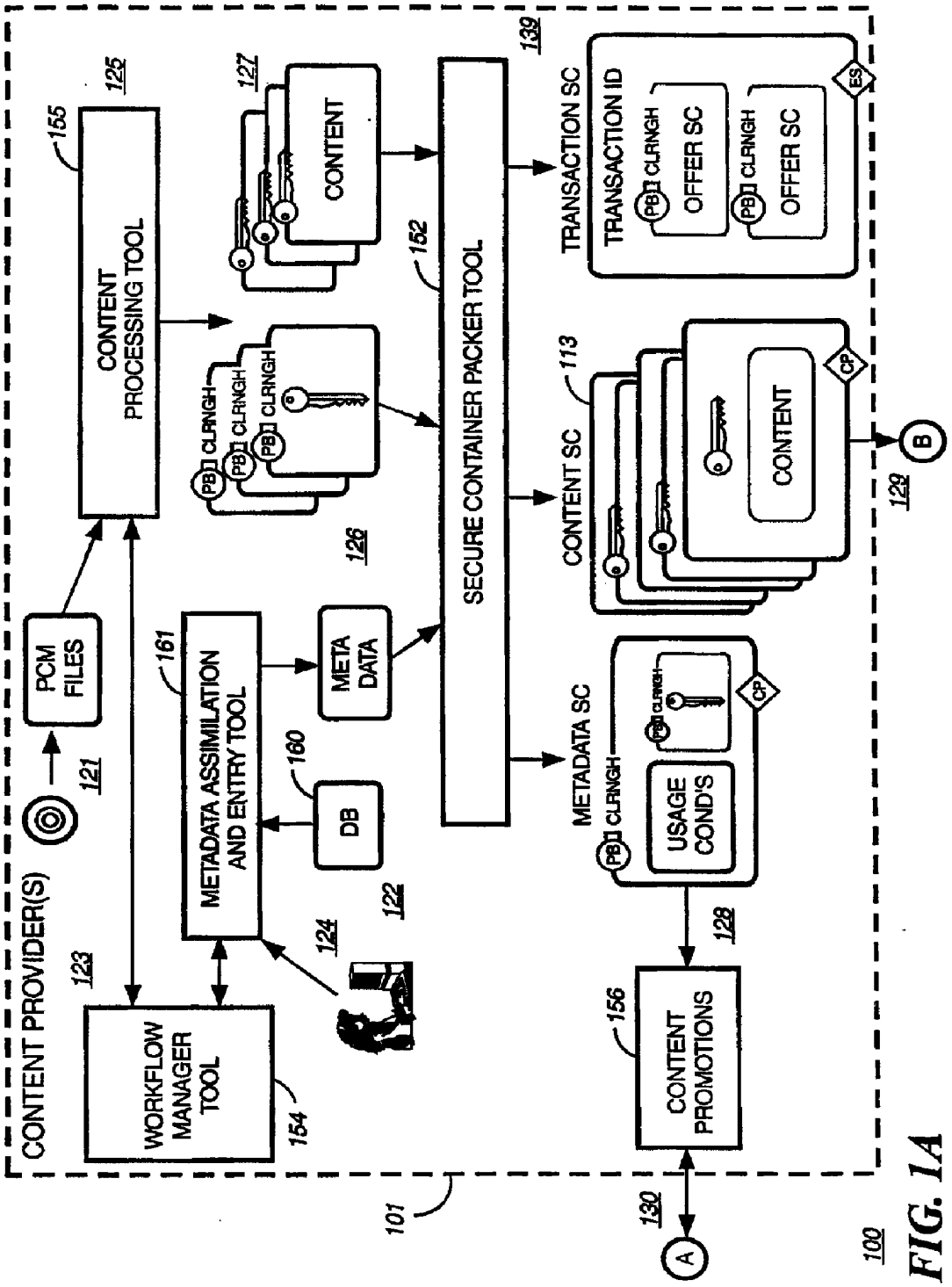

CONTENT PROVIDER(S)
121
PCM FILES
123
WORKFLOW MANAGER TOOL
154
CONTENT PROCESSING TOOL
155
125
METADATA ASSIMILATION AND ENTRY TOOL
161
160
DB
122
124
META DATA
126
CLRNGH
CLRNGH
127
CONTENT
SECURE CONTAINER PACKER TOOL
152
139
TRANSACTION SC
TRANSACTION ID
CLRNGH
OFFER SC
CLRNGH
OFFER SC
ES
CONTENT SC
113
CONTENT
CP
129
B
METADATA SC
CLRNGH
USAGE COND'S
CP
128
156
CONTENT PROMOTIONS
130
A
101
100

FIG. 1A

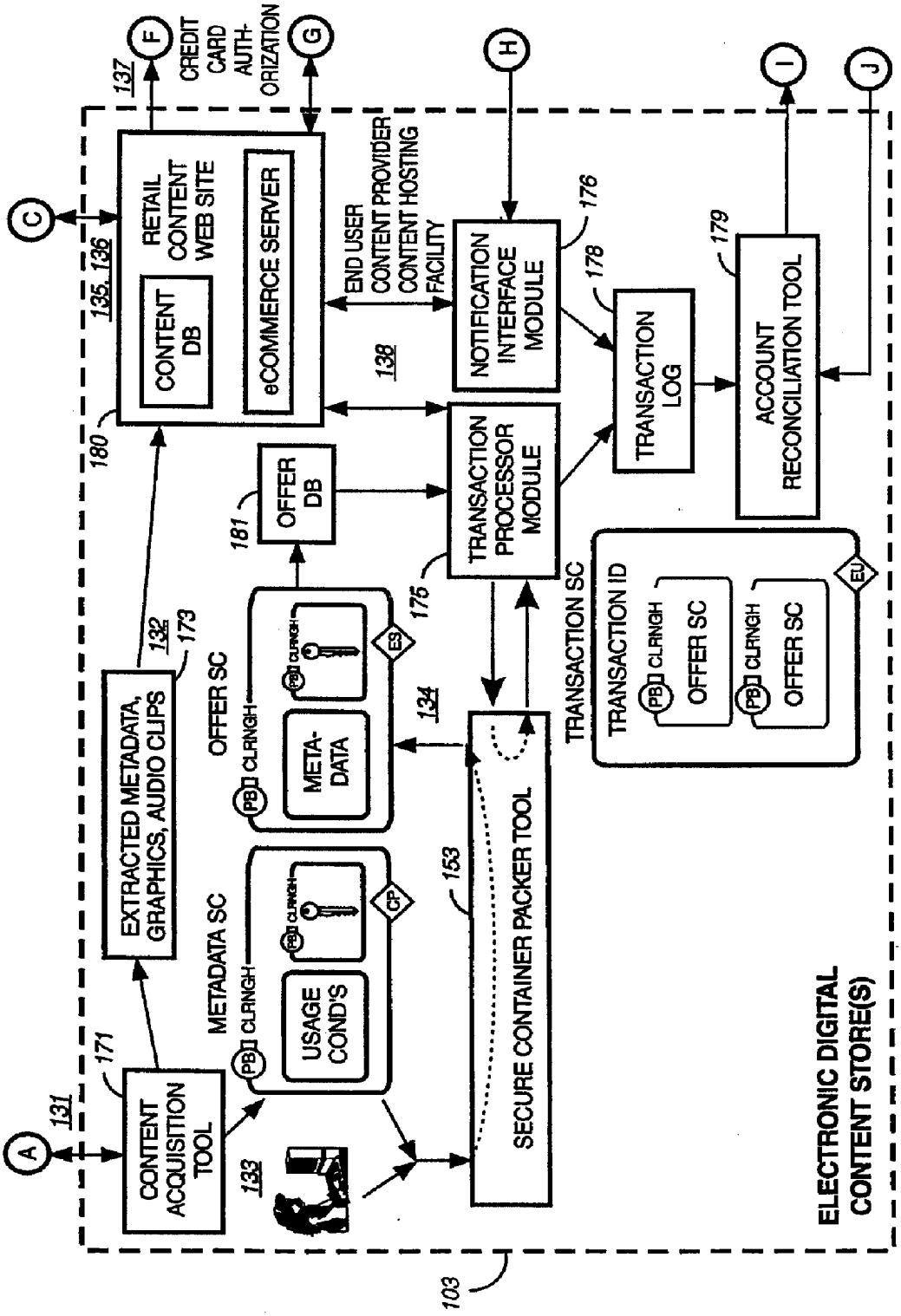

ELECTRONIC DIGITAL CONTENT STORE(S)
103
CONTENT ACQUISITION TOOL
131
171
EXTRACTED METADATA, GRAPHICS, AUDIO CLIPS
132
173
METADATA SC
USAGE COND'S
OFFER SC
META-DATA
SECURE CONTAINER PACKER TOOL
153
134
TRANSACTION SC
TRANSACTION ID
OFFER SC
OFFER SC
TRANSACTION PROCESSOR MODULE
175
OFFER DB
181
RETAIL CONTENT WEB SITE
135, 136
CONTENT DB
eCOMMERCE SERVER
180
END USER CONTENT PROVIDER CONTENT HOSTING FACILITY
138
NOTIFICATION INTERFACE MODULE
176
178
TRANSACTION LOG
179
ACCOUNT RECONCILIATION TOOL
137
CREDIT CARD AUTH-ORIZATION
A
C
F
G
H
I
J

FIG. 1B 100

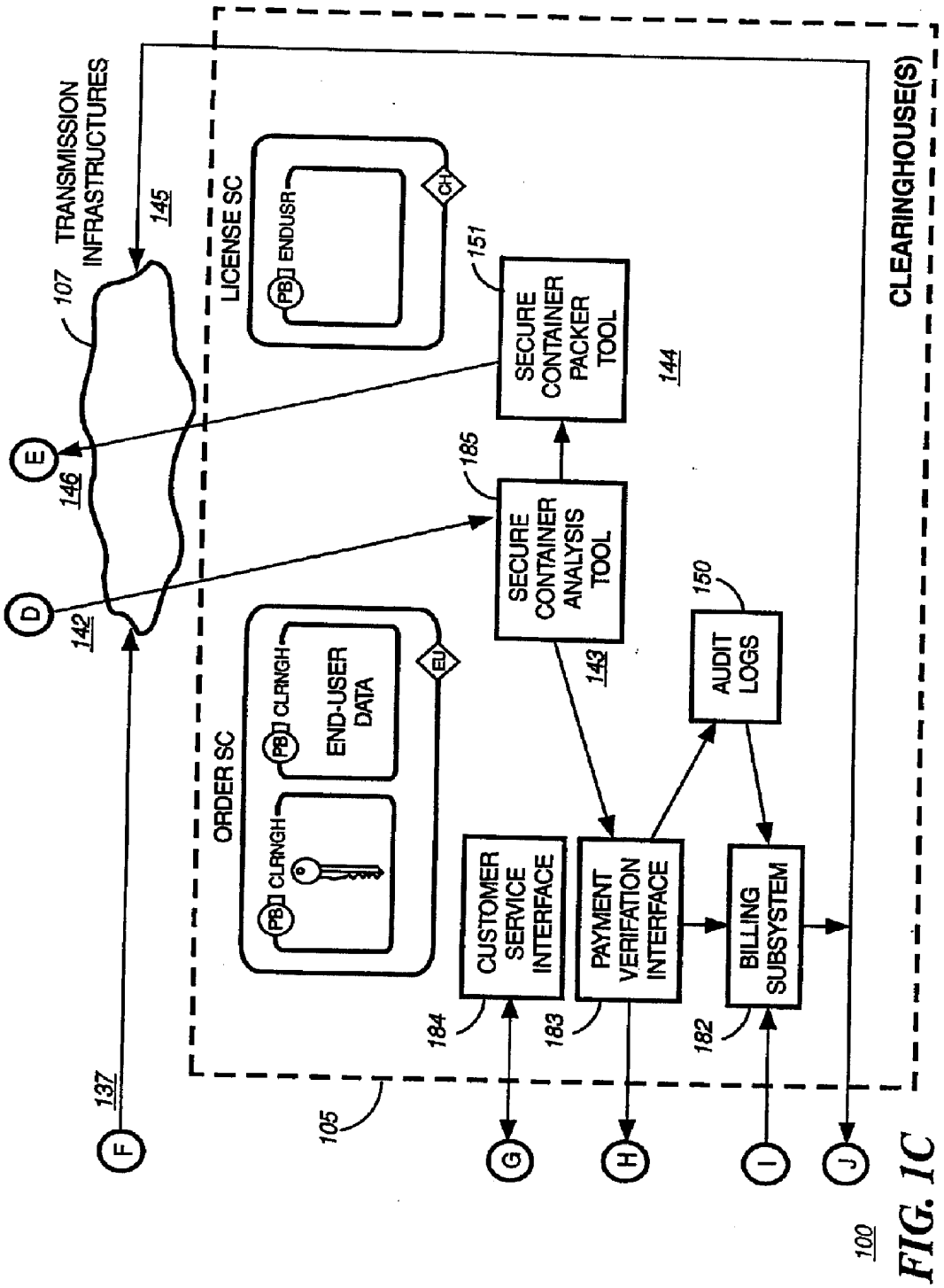

CLEARINGHOUSE(S)
TRANSMISSION INFRASTRUCTURES
107
145
LICENSE SC
ENDUSR
CH
PB
151
SECURE CONTAINER PACKER TOOL
144
185
SECURE CONTAINER ANALYSIS TOOL
143
150
AUDIT LOGS
ORDER SC
END-USER DATA
CLRNGH
EU
CUSTOMER SERVICE INTERFACE
PAYMENT VERIFATION INTERFACE
BILLING SUBSYSTEM
184
183
182
105
137
142
146
D
E
F
G
H
I
J
100

FIG. 1C

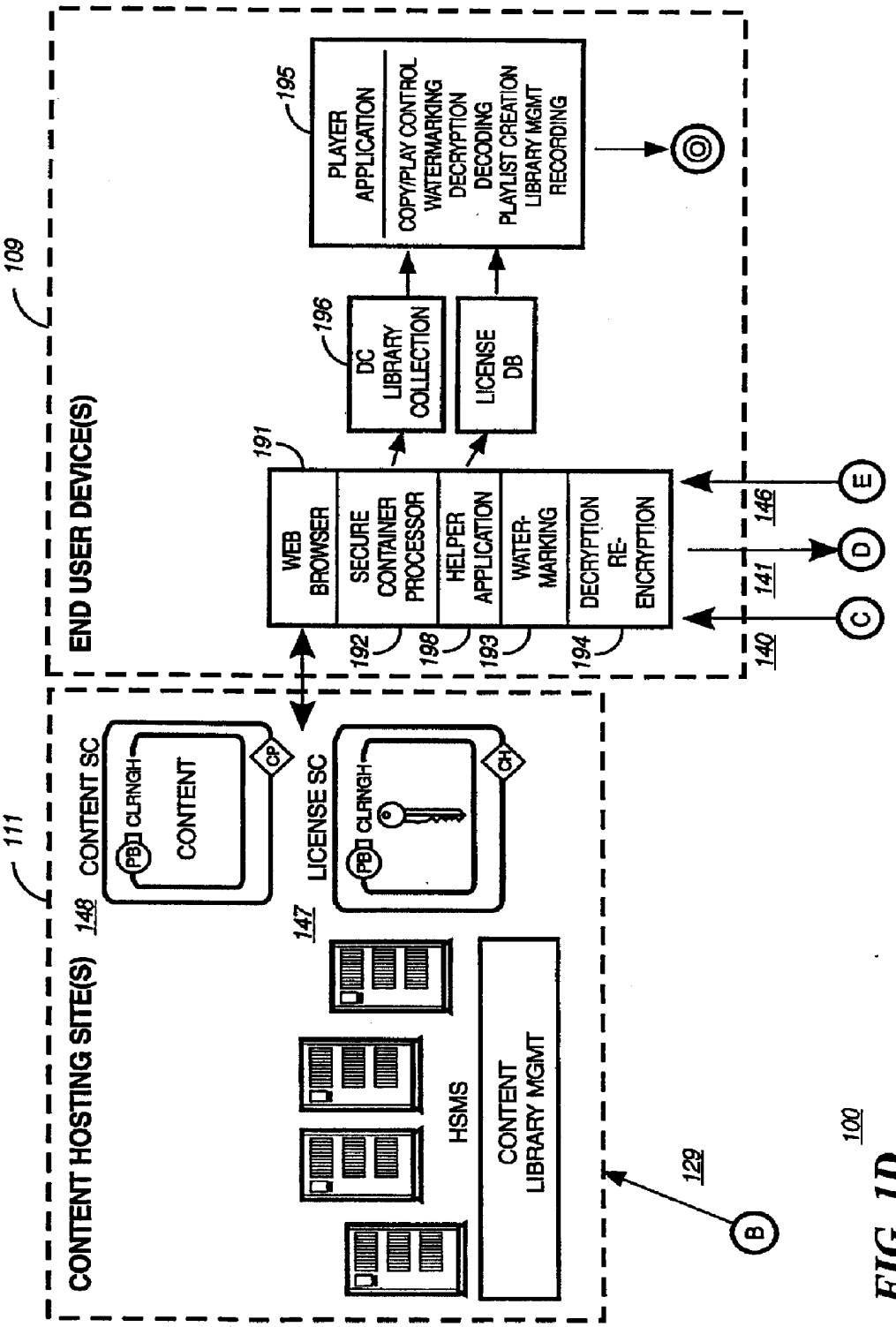

109
END USER DEVICE(S)
195
PLAYER APPLICATION
COPY/PLAY CONTROL
WATERMARKING
DECRYPTION
DECODING
PLAYLIST CREATION
LIBRARY MGMT
RECORDING
196
DC LIBRARY COLLECTION
LICENSE DB
191
WEB BROWSER
SECURE CONTAINER PROCESSOR
HELPER APPLICATION
WATER-MARKING
DECRYPTION RE-ENCRYPTION
192
198
193
194
140
146
141
E
D
C
111
CONTENT HOSTING SITE(S)
148
CONTENT SC
CLRNGH
CONTENT
CP
LICENSE SC
CLRNGH
CH
147
HSMS
CONTENT LIBRARY MGMT
129
B
100

FIG. 1D

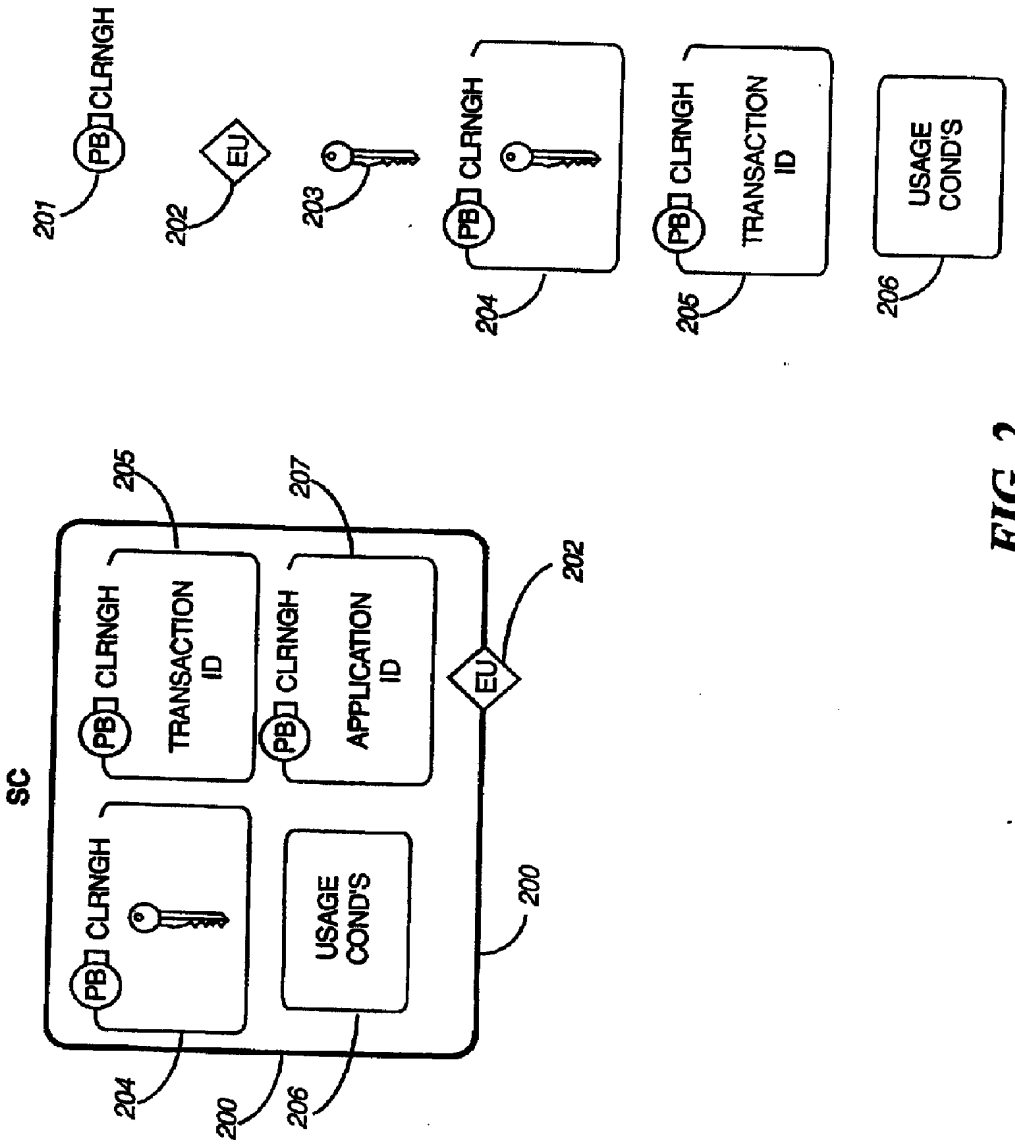

201
PB
CLRNGH
202
EU
203
204
PB
CLRNGH
205
PB
CLRNGH
TRANSACTION ID
206
USAGE COND'S
SC
205
207
PB
CLRNGH
TRANSACTION ID
PB
CLRNGH
APPLICATION ID
EU
202
PB
CLRNGH
USAGE COND'S
200
204
200
206

FIG. 2

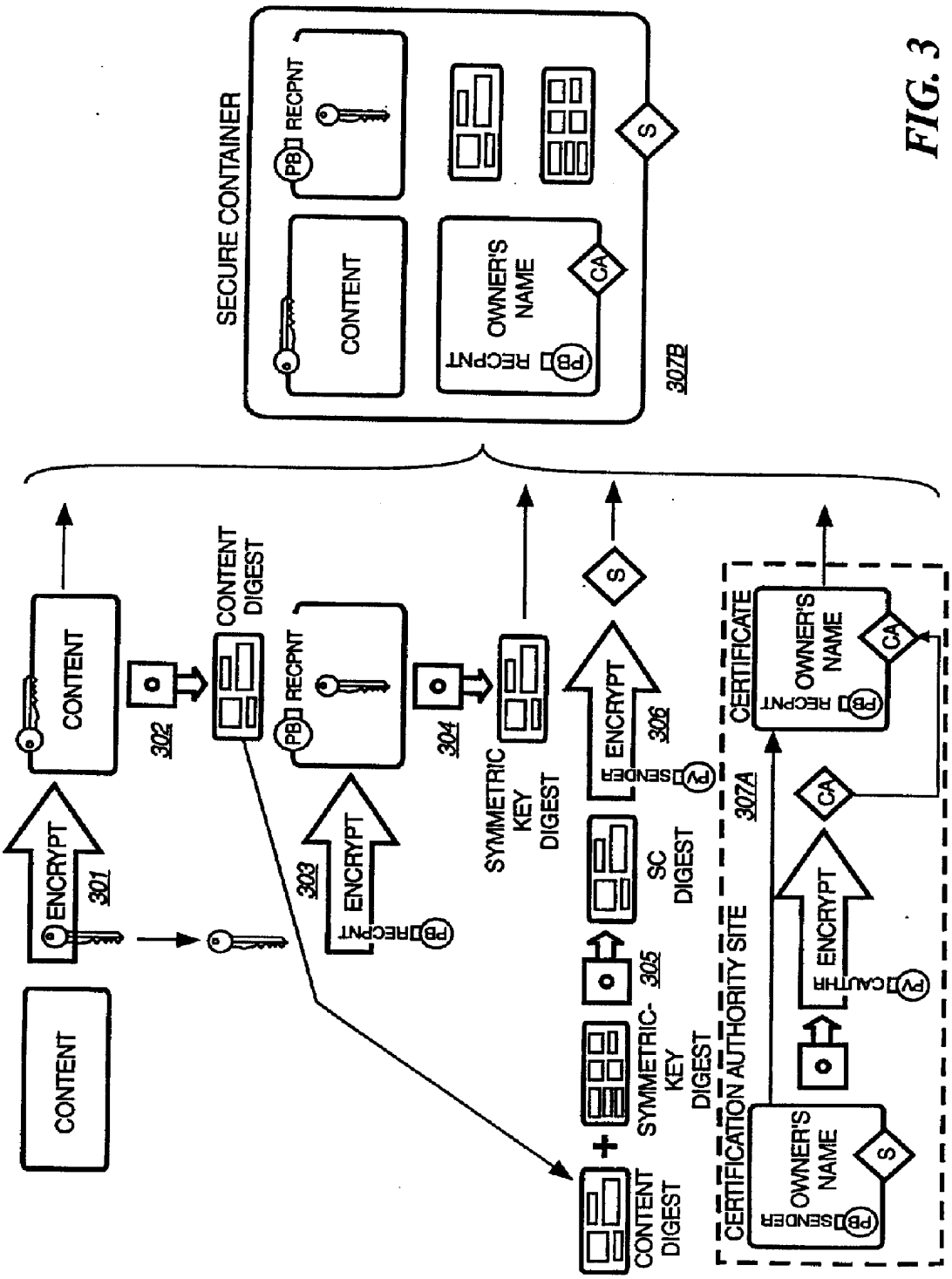

SECURE CONTAINER
RECPNT
CONTENT
OWNER'S NAME
CA
S
307B
CONTENT
ENCRYPT
301
302
CONTENT DIGEST
RECPNT
303
ENCRYPT
304
SYMMETRIC KEY DIGEST
ENCRYPT
306
SENDER
SC DIGEST
305
SYMMETRIC-KEY DIGEST
CONTENT DIGEST
CERTIFICATE
OWNER'S NAME
CA
307A
CERTIFICATION AUTHORITY SITE
ENCRYPT
CAUTHR
OWNER'S NAME
S
SENDER

FIG. 3

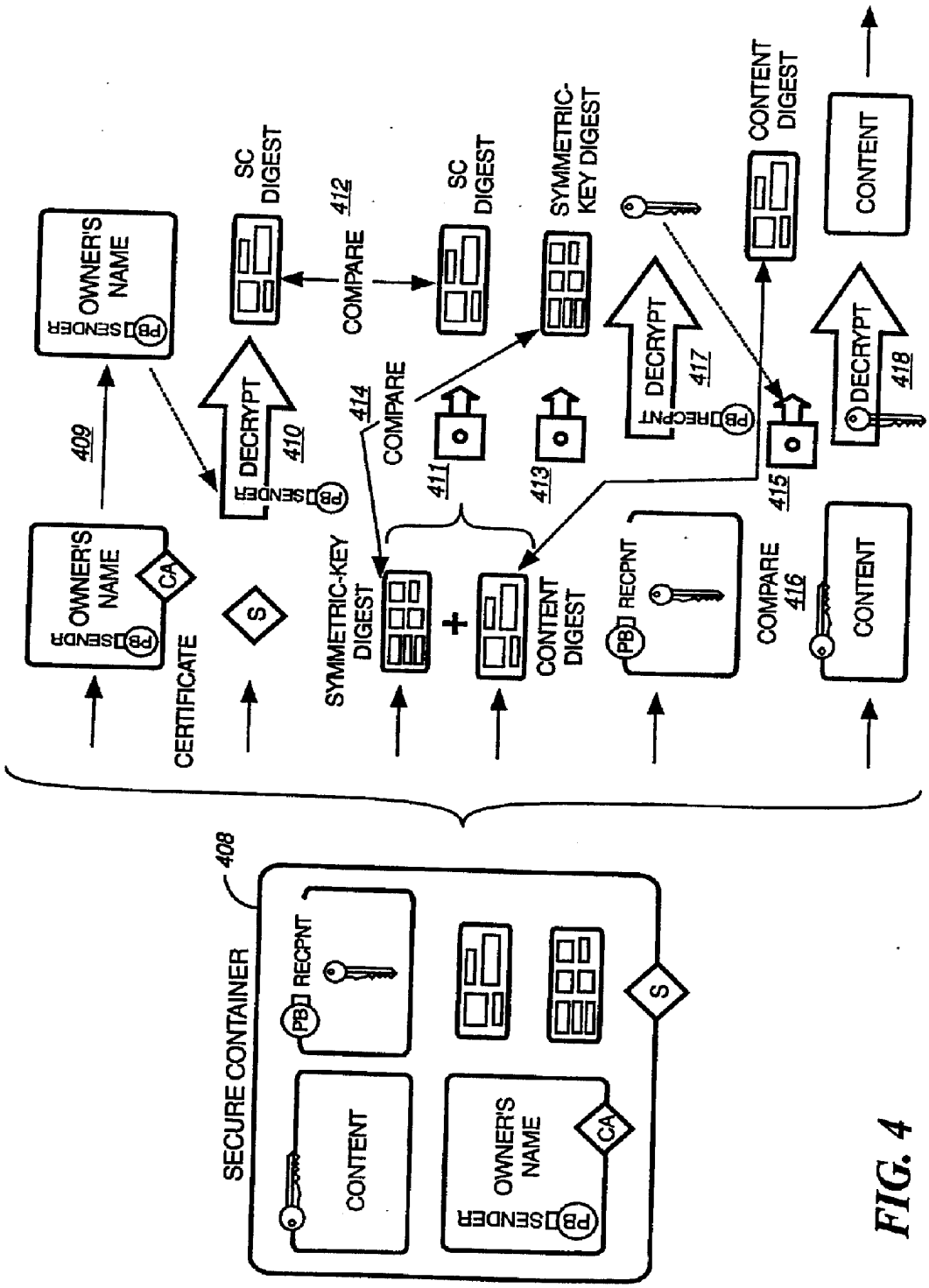

SECURE CONTAINER
408
CONTENT
RECPNT
OWNER'S NAME
SENDER
CA
S
409
CERTIFICATE
DECRYPT
410
SC DIGEST
COMPARE
412
SC DIGEST
SYMMETRIC-KEY DIGEST
411
413
414
COMPARE
SYMMETRIC-KEY DIGEST
CONTENT DIGEST
DECRYPT
417
CONTENT DIGEST
415
COMPARE
416
DECRYPT
418
CONTENT

FIG. 4

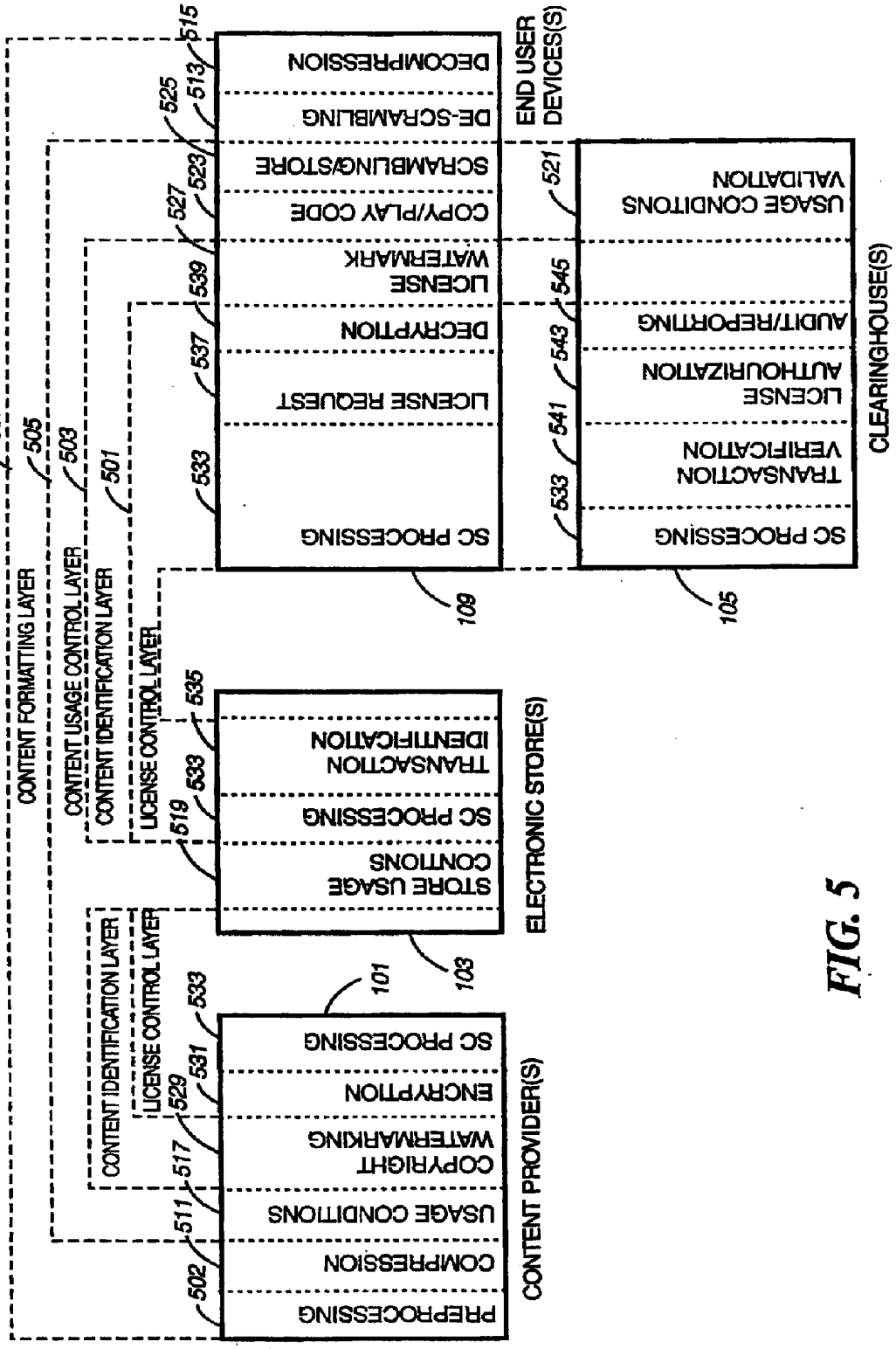

CONTENT FORMATTING LAYER
CONTENT USAGE CONTROL LAYER
CONTENT IDENTIFICATION LAYER
LICENSE CONTROL LAYER
CONTENT IDENTIFICATION LAYER
LICENSE CONTROL LAYER
PREPROCESSING
COMPRESSION
USAGE CONDITIONS
COPYRIGHT WATERMARKING
ENCRYPTION
SC PROCESSING
CONTENT PROVIDER(S)
STORE USAGE CONTIONS
SC PROCESSING
TRANSACTION IDENTIFICATION
ELECTRONIC STORE(S)
SC PROCESSING
LICENSE REQUEST
DECRYPTION
LICENSE WATERMARK
COPY/PLAY CODE
SCRAMBLING/STORE
DE-SCRAMBLING
DECOMPRESSION
END USER DEVICES(S)
SC PROCESSING
TRANSACTION VERIFICATION
LICENSE AUTHOURIZATION
AUDIT/REPORTING
USAGE CONDITIONS VALIDATION
CLEARINGHOUSE(S)
101
103
105
109
501
502
503
505
507
511
513
515
517
519
521
523
525
527
529
531
533
535
537
539
541
543
545

FIG. 5

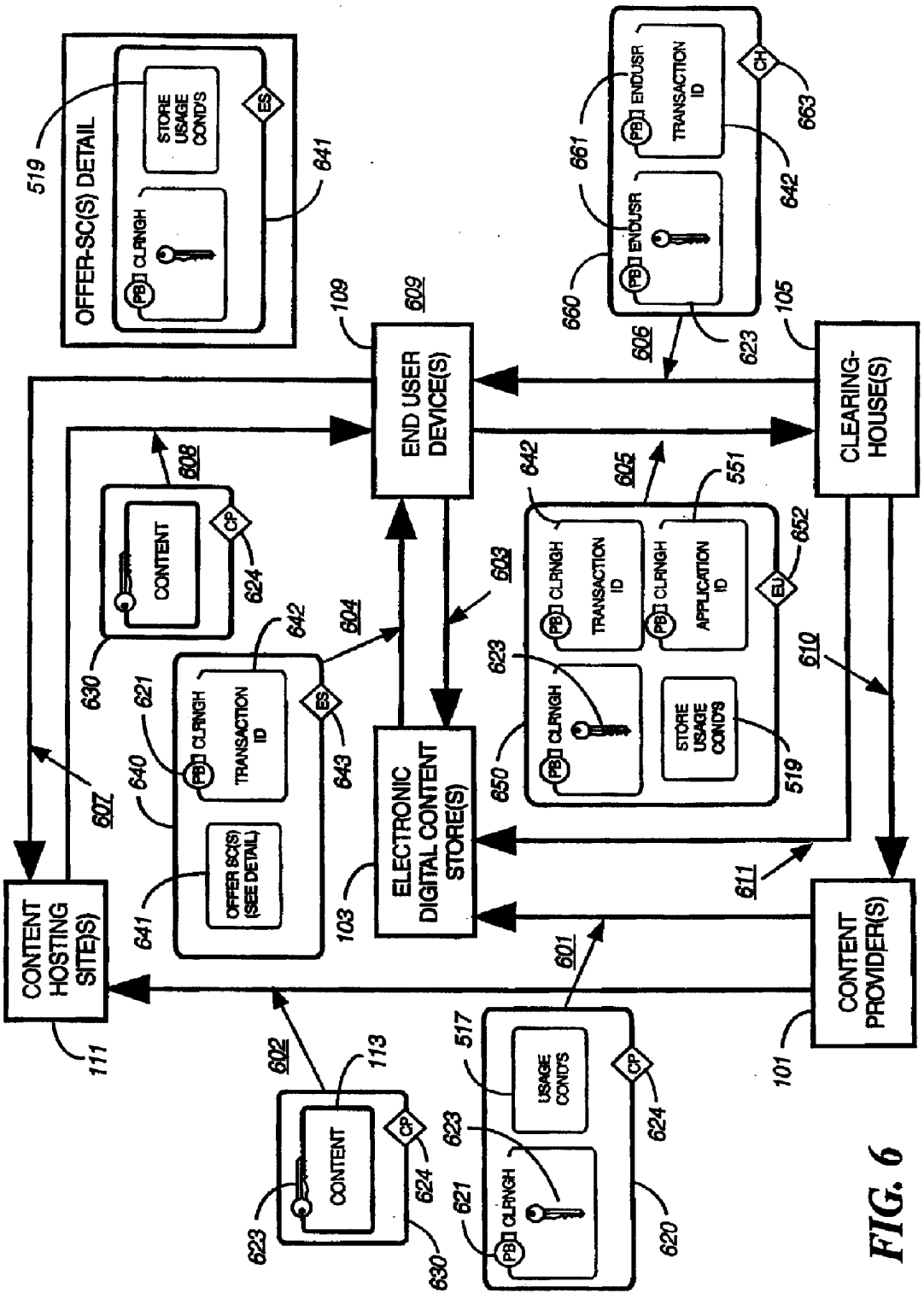

OFFER-SC(S) DETAIL
519
STORE USAGE COND'S
CLRNGH
641
END USER DEVICE(S)
109
609
CLEARING-HOUSE(S)
105
606
660
661
ENDUSR
TRANSACTION ID
642
623
663
CONTENT
608
630
624
605
642
551
652
603
CLRNGH
TRANSACTION ID
APPLICATION ID
623
610
650
519
604
621
640
643
607
OFFER SC(S) (SEE DETAIL)
641
ELECTRONIC DIGITAL CONTENT STORE(S)
103
611
CONTENT HOSTING SITE(S)
111
601
CONTENT PROVIDER(S)
101
602
113
517
USAGE COND'S
620

FIG. 6

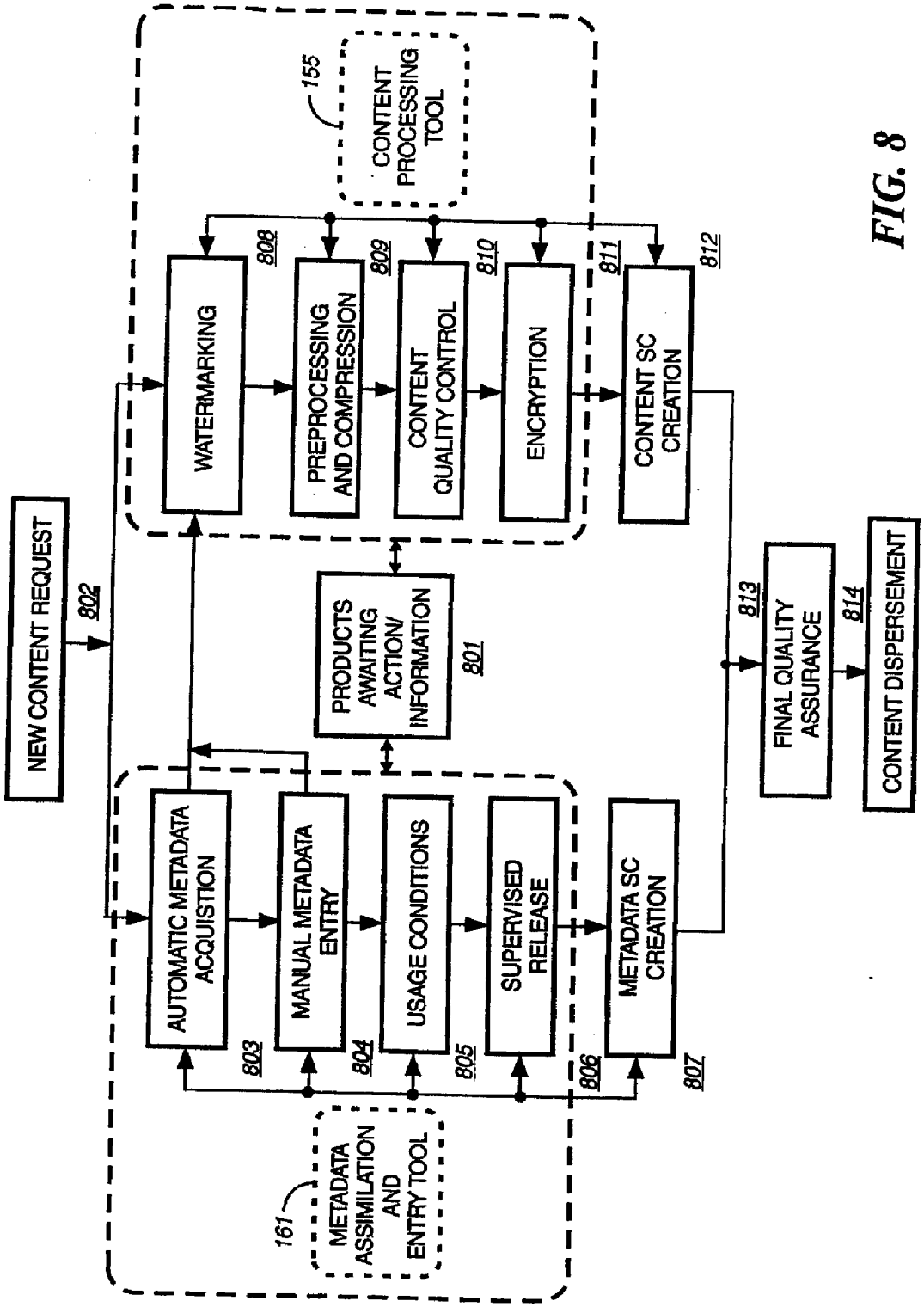

NEW CONTENT REQUEST
802
WATERMARKING
808
PREPROCESSING AND COMPRESSION
809
CONTENT QUALITY CONTROL
810
ENCRYPTION
811
CONTENT SC CREATION
812
155
CONTENT PROCESSING TOOL
PRODUCTS AWAITING ACTION/ INFORMATION
801
AUTOMATIC METADATA ACQUISITION
803
MANUAL METADATA ENTRY
804
USAGE CONDITIONS
805
SUPERVISED RELEASE
806
METADATA SC CREATION
807
161
METADATA ASSIMILATION AND ENTRY TOOL
FINAL QUALITY ASSURANCE
813
814
CONTENT DISPERSEMENT

FIG. 8

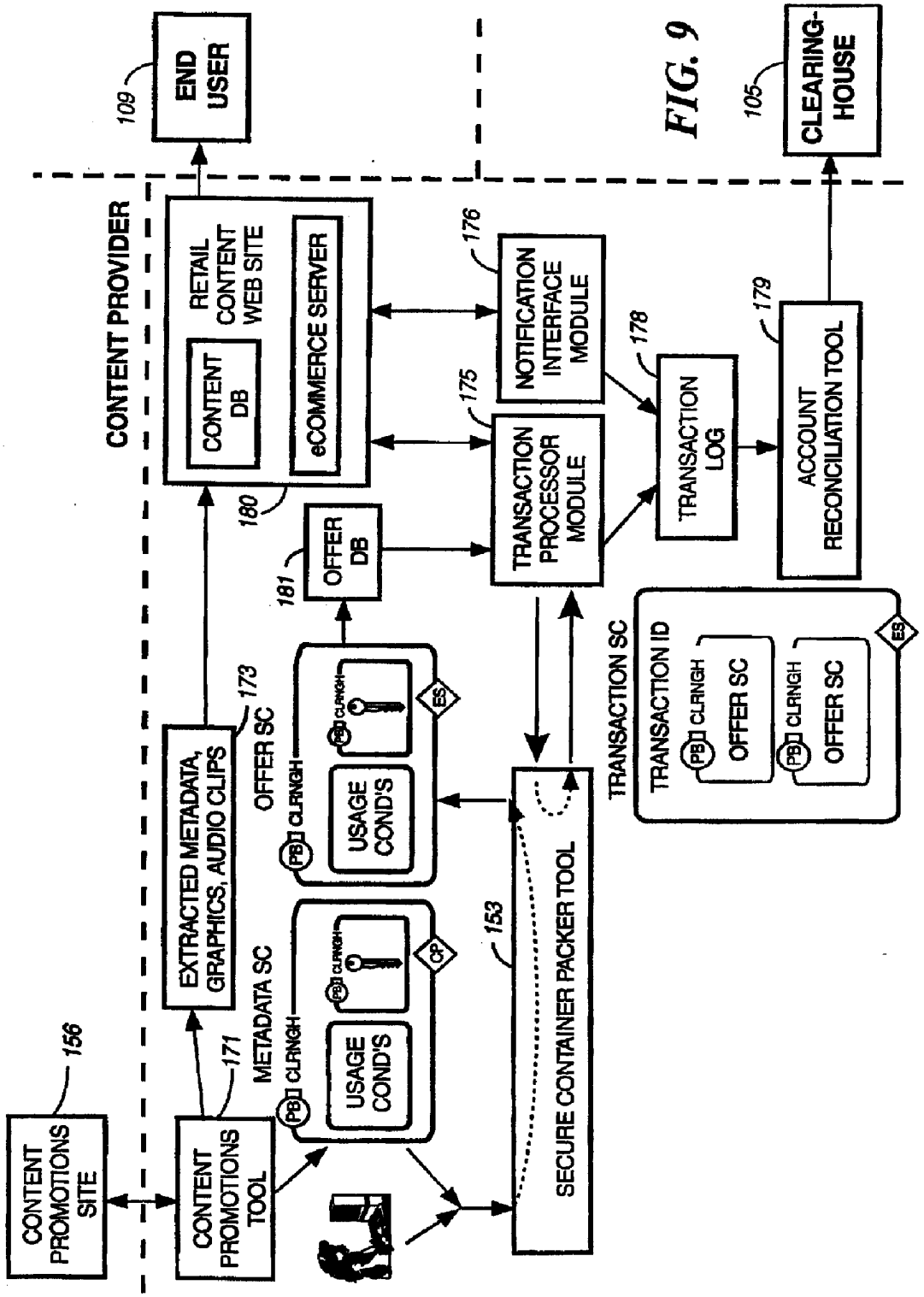

END USER
109
CLEARING-HOUSE
105
CONTENT PROVIDER
RETAIL CONTENT WEB SITE
CONTENT DB
eCOMMERCE SERVER
180
176
NOTIFICATION INTERFACE MODULE
175
TRANSACTION PROCESSOR MODULE
178
TRANSACTION LOG
179
ACCOUNT RECONCILIATION TOOL
OFFER DB
181
EXTRACTED METADATA, GRAPHICS, AUDIO CLIPS
173
OFFER SC
USAGE COND'S
CLRNGH
PB
ES
METADATA SC
CP
153
SECURE CONTAINER PACKER TOOL
TRANSACTION SC
TRANSACTION ID
OFFER SC
CONTENT PROMOTIONS SITE
156
CONTENT PROMOTIONS TOOL
171

FIG. 9

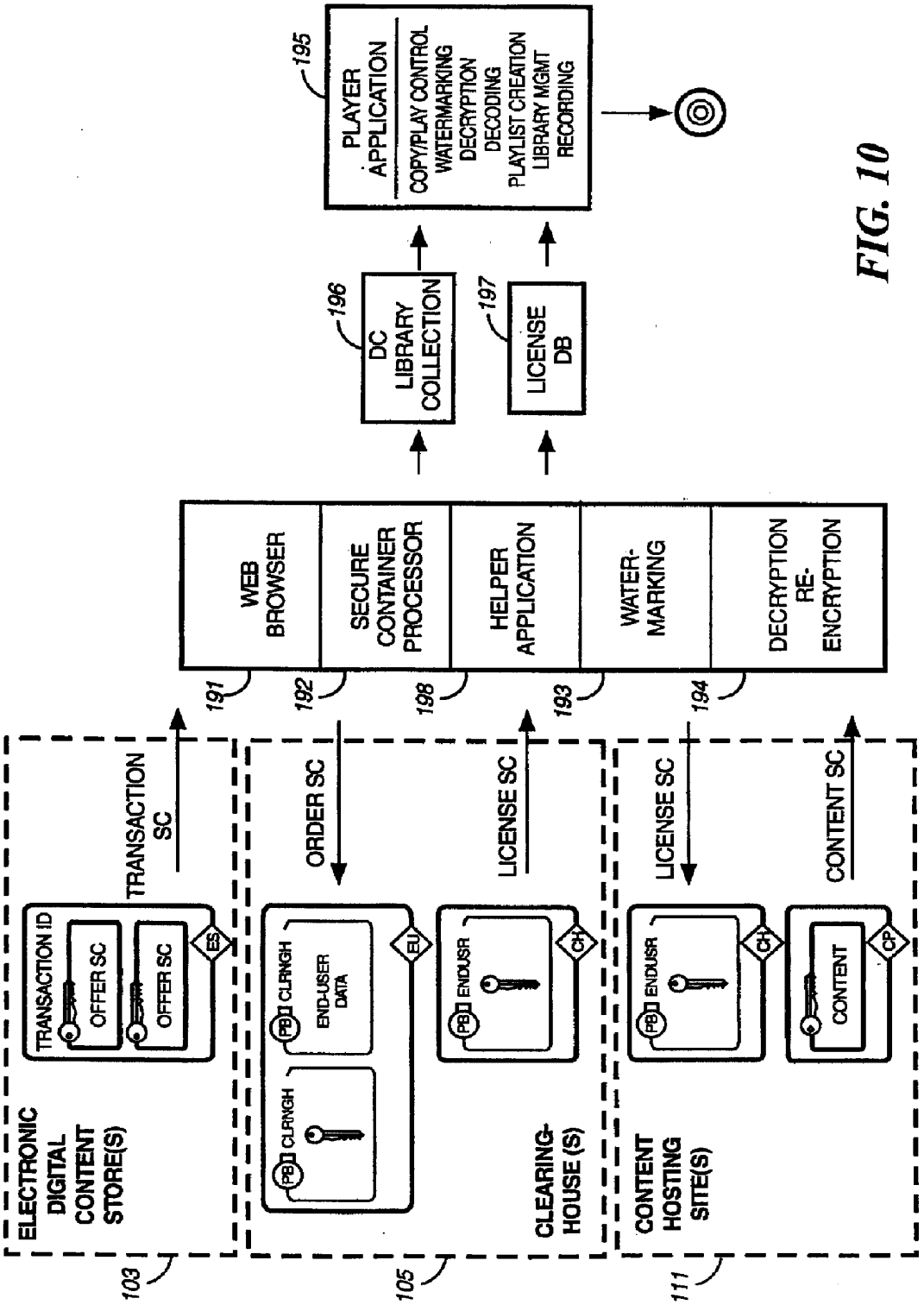

PLAYER APPLICATION
COPY/PLAY CONTROL
WATERMARKING
DECRYPTION
DECODING
PLAYLIST CREATION
LIBRARY MGMT
RECORDING
195
DC LIBRARY COLLECTION
196
LICENSE DB
197
WEB BROWSER
SECURE CONTAINER PROCESSOR
HELPER APPLICATION
WATER-MARKING
DECRYPTION RE-ENCRYPTION
191
192
198
193
194
TRANSACTION SC
ORDER SC
LICENSE SC
LICENSE SC
CONTENT SC
ELECTRONIC DIGITAL CONTENT STORE(S)
TRANSACTION ID
OFFER SC
OFFER SC
ES
CLEARING-HOUSE (S)
CLRNGH
END-USER DATA
EU
CLRNGH
ENDUSR
CH
CONTENT HOSTING SITE(S)
ENDUSR
CH
CONTENT
CP
103
105
111

FIG. 10

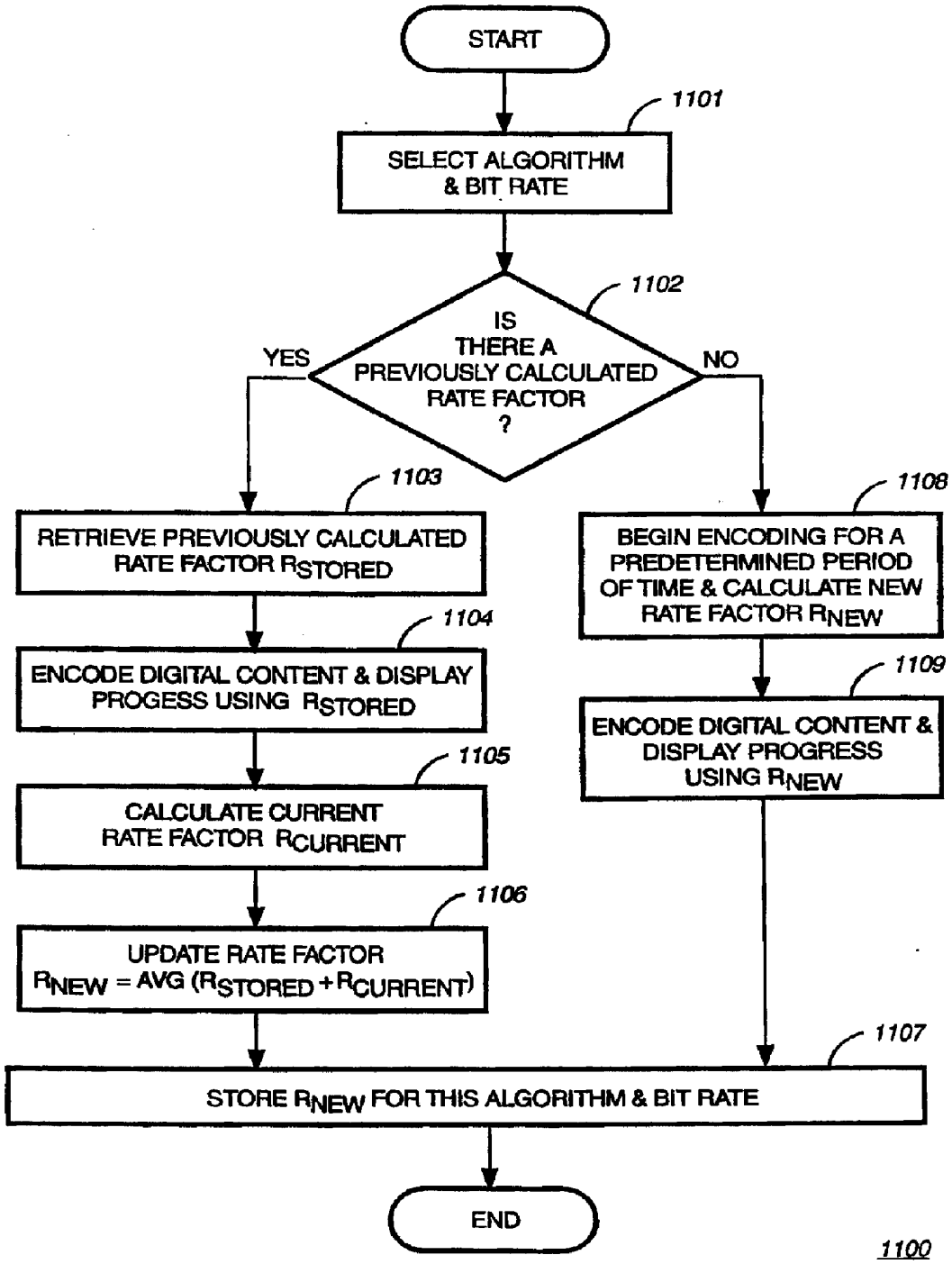

START
1101
SELECT ALGORITHM & BIT RATE
1102
IS THERE A PREVIOUSLY CALCULATED RATE FACTOR ?
YES
NO
1103
RETRIEVE PREVIOUSLY CALCULATED RATE FACTOR RSTORED
1104
ENCODE DIGITAL CONTENT & DISPLAY PROGESS USING RSTORED
1105
CALCULATE CURRENT RATE FACTOR RCURRENT
1106
UPDATE RATE FACTOR RNEW = AVG (RSTORED + RCURRENT)
1108
BEGIN ENCODING FOR A PREDETERMINED PERIOD OF TIME & CALCULATE NEW RATE FACTOR RNEW
1109
ENCODE DIGITAL CONTENT & DISPLAY PROGRESS USING RNEW
1107
STORE RNEW FOR THIS ALGORITHM & BIT RATE
END
1100

FIG. 11

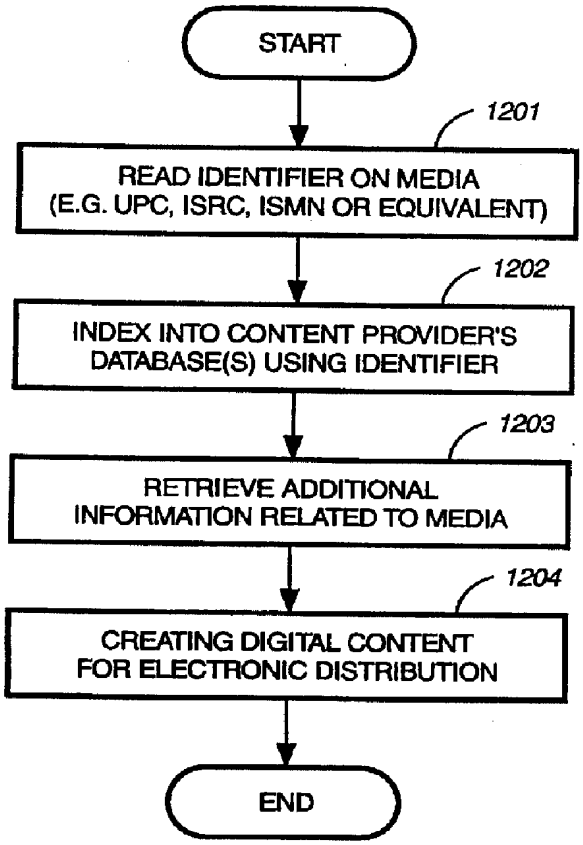

START
1201
READ IDENTIFIER ON MEDIA
(E.G. UPC, ISRC, ISMN OR EQUIVALENT)
1202
INDEX INTO CONTENT PROVIDER'S
DATABASE(S) USING IDENTIFIER
1203
RETRIEVE ADDITIONAL
INFORMATION RELATED TO MEDIA
1204
CREATING DIGITAL CONTENT
FOR ELECTRONIC DISTRIBUTION
END

FIG. 12

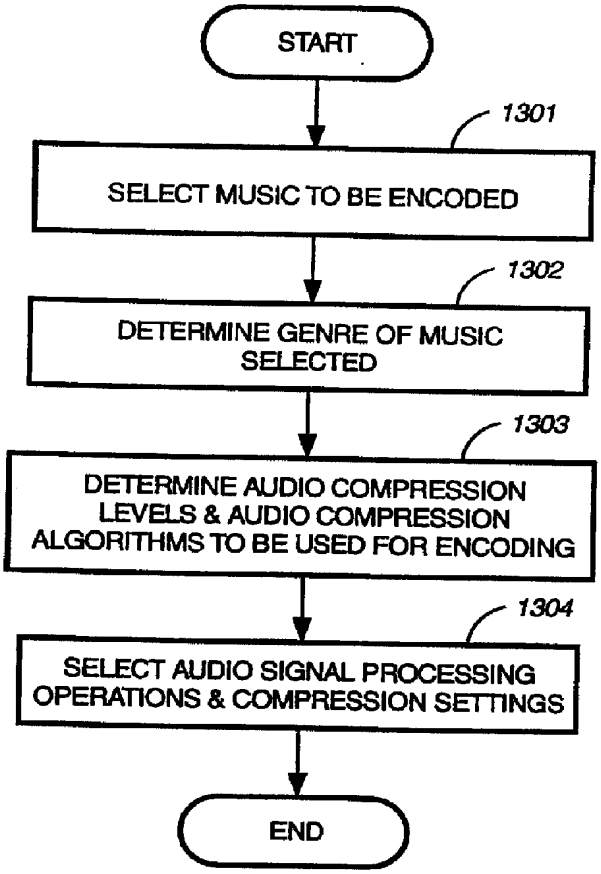

START
1301
SELECT MUSIC TO BE ENCODED
1302
DETERMINE GENRE OF MUSIC SELECTED
1303
DETERMINE AUDIO COMPRESSION LEVELS & AUDIO COMPRESSION ALGORITHMS TO BE USED FOR ENCODING
1304
SELECT AUDIO SIGNAL PROCESSING OPERATIONS & COMPRESSION SETTINGS
END

FIG. 13