

Petitioner's Demonstratives

Google LLC
v.
Sandpiper CDN, LLC

IPR2025-01010
U.S. Patent No. 10,057,322

September 1, 2026

Overview - '322 Patent



US010057322B2

(12) **United States Patent**
Newton et al.

(10) **Patent No.:** US 10,057,322 B2
(45) **Date of Patent:** Aug. 21, 2018

(54) **NETWORK ADDRESS RESOLUTION**

(71) Applicant: **Level 3 Communications, LLC**,
Broomfield, CO (US)

(72) Inventors: **Christopher Newton**, Westlake Village,
CA (US); **William R. Power**, Boulder,
CO (US)

(73) Assignee: **Level 3 Communications, LLC**,
Broomfield, CO (US)

(*) Notice: Subject to any disclaimer, the term of this
patent is extended or adjusted under 35
U.S.C. 154(b) by 290 days.

(21) Appl. No.: **14/985,968**

(22) Filed: **Dec. 31, 2015**

(65) **Prior Publication Data**

US 2016/0191455 A1 Jun. 30, 2016

Related U.S. Application Data

(60) Provisional application No. 62/098,930, filed on Dec.
31, 2014.

(51) **Int. Cl.**

H04L 29/08 (2006.01)
H04L 29/12 (2006.01)
G06F 17/50 (2006.01)

(52) **U.S. Cl.**

CPC **H04L 67/02** (2013.01); **G06F 17/30038**
(2013.01); **G06F 17/30876** (2013.01); **H04L**
61/1511 (2013.01); **H04L 61/3055** (2013.01)

(58) **Field of Classification Search**

CPC **H04L 67/02**; **H04L 67/1021**
See application file for complete search history.

U.S. PATENT DOCUMENTS

6,185,598 B1	2/2001	Farber et al.	
6,654,807 B2	11/2003	Farber et al.	
7,054,935 B2	5/2006	Farber et al.	
7,945,693 B2	5/2011	Farber et al.	
7,940,779 B2	5/2011	Farber et al.	
8,060,613 B2	11/2011	Farber et al.	
8,250,211 B2	8/2012	Swidens et al.	
8,281,035 B2	10/2012	Farber et al.	
8,291,046 B2	10/2012	Farber et al.	
8,296,396 B2	10/2012	Farber et al.	
8,412,823 B1 *	4/2013	Richardson	H04L 47/70 709/225
8,463,877 B1 *	6/2013	Richardson	H04L 67/327 709/219
8,468,245 B2	6/2013	Farber et al.	
8,473,613 B2	6/2013	Farber et al.	

(Continued)

OTHER PUBLICATIONS

International Preliminary Report on Patentability dated Jul. 4, 2017,
Int'l Appl. No. PCT/US15/068284, Int'l Filing Date Dec. 31, 2015
9 pgs.

(Continued)

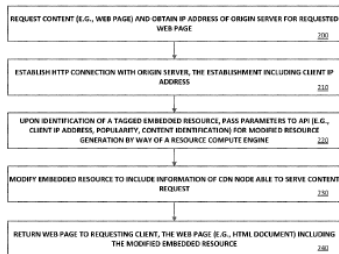
Primary Examiner — Lashonda Jacobs

(57)

ABSTRACT

A content delivery method, and related apparatus, that
involves the operations of receiving a request for a content
resource including at least one embedded resource with a
tag. Upon identification of the tag, using at least one delivery
parameter to generate a modified embedded resource.
Embedding the modified embedded resource in a content
resource, such as an HTML document. Delivering, such
as in response to request from a browser, the content
resource with the modified embedded resource.

15 Claims, 6 Drawing Sheets



Network Address Resolution

Earliest priority claim: December 31, 2014

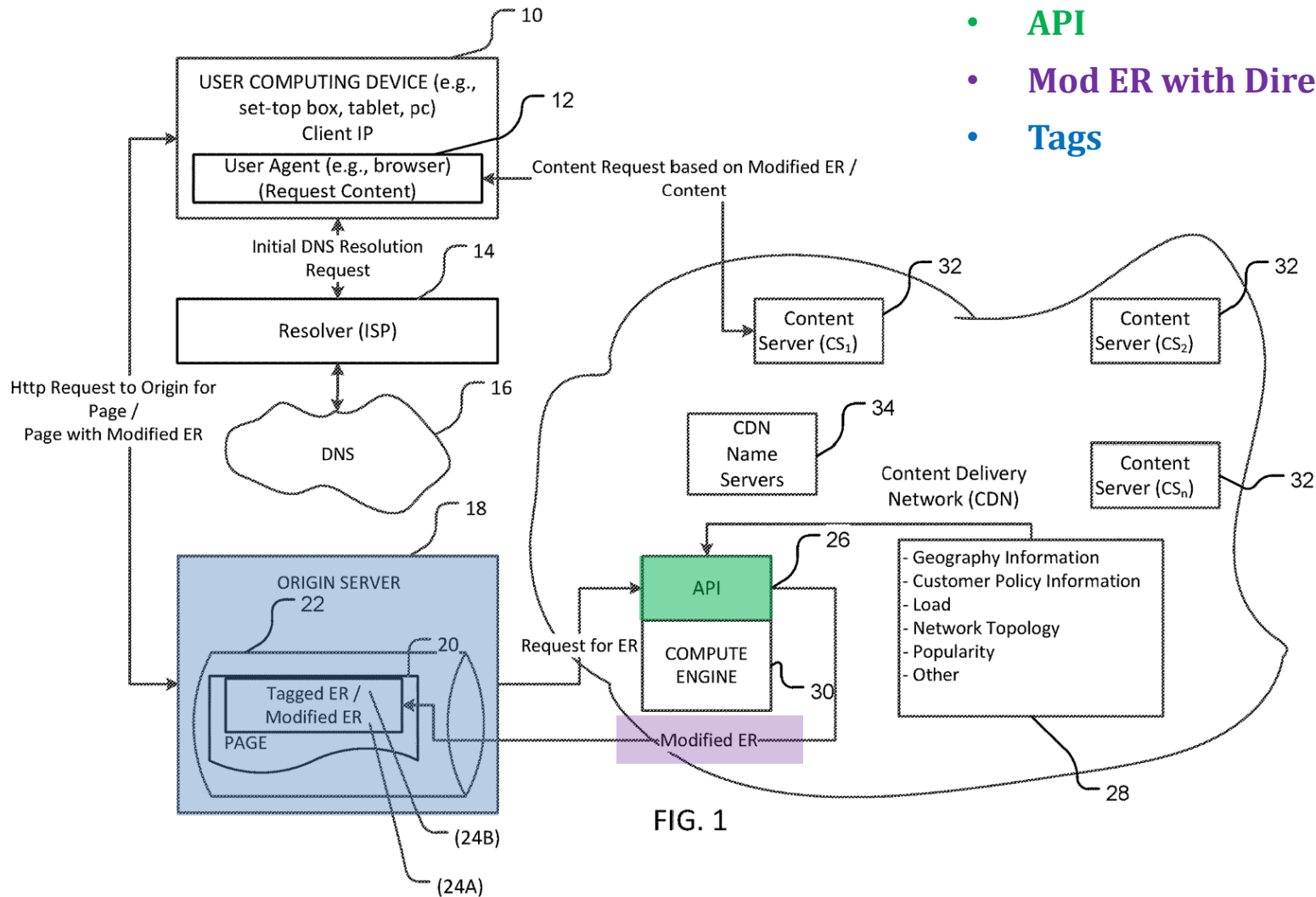
15 Claims (1, 11, and 15 independent)

Grounds Presented

Ground	Claims Challenged (independent in bold)	Basis	References
1A	15	102	<i>Verma</i>
1B	11, 12, 14, 15	103	<i>Verma</i>
2	11, 12, 14, 15	103	<i>Verma and Raciborski</i>
3	1, 8, 10, 13	103	<i>Verma and Lewin</i>
4	2, 4, 5, 7, 9, 13	103	<i>Verma, Raciborski, and Lewin</i>
5	3, 6	103	<i>Verma, Raciborski, Lewin, and Drai</i>

Overview - '322 Patent

'322 Patent, Figure 1 (annotated)



Disputed Limitations

- **API**
- **Mod ER with Direct Link**
- **Tags**

Challenged Claims & Disputed Limitations

Independent Claim 15 (Representative of Independent Claim 11)*	Shorthand
A method comprising:	[15pre]
providing, at a content delivery network, an application programming interface (API);	[15a] API
at the API, receiving, from an origin system that is separate from the content delivery network, an embedded resource and at least one parameter associated with the embedded resource;	[15b]
generating a modified embedded resource providing a direct link to a node within the content delivery network for obtaining content associated with the embedded resource;	[15c] Mod ER with direct link
delivering, by the content delivery network, the modified embedded resource to the origin system;	[15d]
receiving, at the node, a request from a client device to obtain the content; and	[15e]
providing the content to the client device.	[15f]

* Claim 11 recites other undisputed limitations (processor, non-transitory computer readable medium, computer readable instructions).

Challenged Claims & Disputed Limitations

Independent Claim 1	Shorthand
A content delivery method comprising:	[1pre]
receiving, by an origin system, a request from a client device for a content resource, the content resource including at least one embedded resource with a tag;	[1a] Content resource including ER with a tag
upon identification of the tag , using at least one delivery parameter to obtain, by the origin system, a modified embedded resource from a content delivery network that is separate from the origin system; and	[1b] Identifying tag
delivering, by the origin system, the content resource to the client device with the modified embedded resource, wherein the modified embedded resource includes a direct link to a node within the content delivery network that can deliver content associated with the embedded resource.	[1c] Mod ER with direct link

Claim Construction

Three Disputed Terms

1. “direct link to a node within the content delivery network” (all claims)
2. “application programming interface (API)” (claims 2-7, 11-15)
3. “with a tag” (claims 1-10)

Direct Link to a Node (All Claims)

“a modified embedded resource providing a direct link to a node within the content delivery network”

Petitioner	PO Infringement Contentions in District Court	PO in IPR
“a link pointing to a specific server”	“a link pointing to [a] specific cache server”	“the specification expressly equates ‘directly’ connecting with bypassing DNS through an embedded IP address” “network address that enables a client device to bypass conventional DNS resolution of a hostname when contacting a CDN node”

Direct link to a node means “a link pointing to a specific server”

'322 Patent

The API returns an absolute URL from which the content may be retrieved, and/or a list of hostnames and/or IP addresses that could be used to construct the URLs at the calling application (operation 330). Similarly, the origin

EX1001 6:50-7:1

Direct Link: Intrinsic evidence describes alternative to DNS for node selection

- Specification: API provides alternative to DNS for edge (node) selection
- Prosecution History: Distinguished prior art using DNS to determine the node

'322 Patent

to serve content to a client. Aspects of the present disclosure involve an alternative to using conventional domain name system (DNS) servers for edge selection by instead calling an API or otherwise querying the CDN when constructing a

EX1001, 3:9-14

Prosecution History

Thus, Richardson requires the client to either (a) communicate with the DNS server of the CDN to determine the node that will be used to deliver the content; or (b) execute translation code to ask the CDN for translation information, which translation information may include the address of the node that will be used to deliver the content. In either case, Richardson does not disclose that either (a) the origin server communicates with the CDN to obtain a direct link to a CDN node to serve the content; or (b) the origin server provides that direct link to the client.

EX1002, 71

Direct Link: Not limited to IP Address

- Claim differentiation confirms that “direct link to a node” is broader than an IP address

'322 Patent, claim 11

generating a modified embedded resource providing a direct link to a node within the content delivery network for obtaining content associated with the embedded resource;

'322 Patent, claim 14

14. The content delivery network of claim 11 wherein the modified embedded resource includes at least one of an IP address to facilitate communication with the node to obtain the content associated with the embedded resource, a virtual

EX1001, 11:1-4, 11:17-21



“The doctrine of claim differentiation is at its strongest in this type of case, where the limitation that is sought to be ‘read into’ an independent claim already appears in a dependent claim.”

InterDigital Commc'ns, LLC v. ITC, 690 F.3d 1318, 1324-25 (Fed. Cir. 2012) (citation and quotations omitted).

Direct Link: Consistent with PO in District Court

- Direct link to a node means “a link pointing to a specific server”
- This is consistent with PO’s statements in district court

PO’s District Court Infringement Contentions

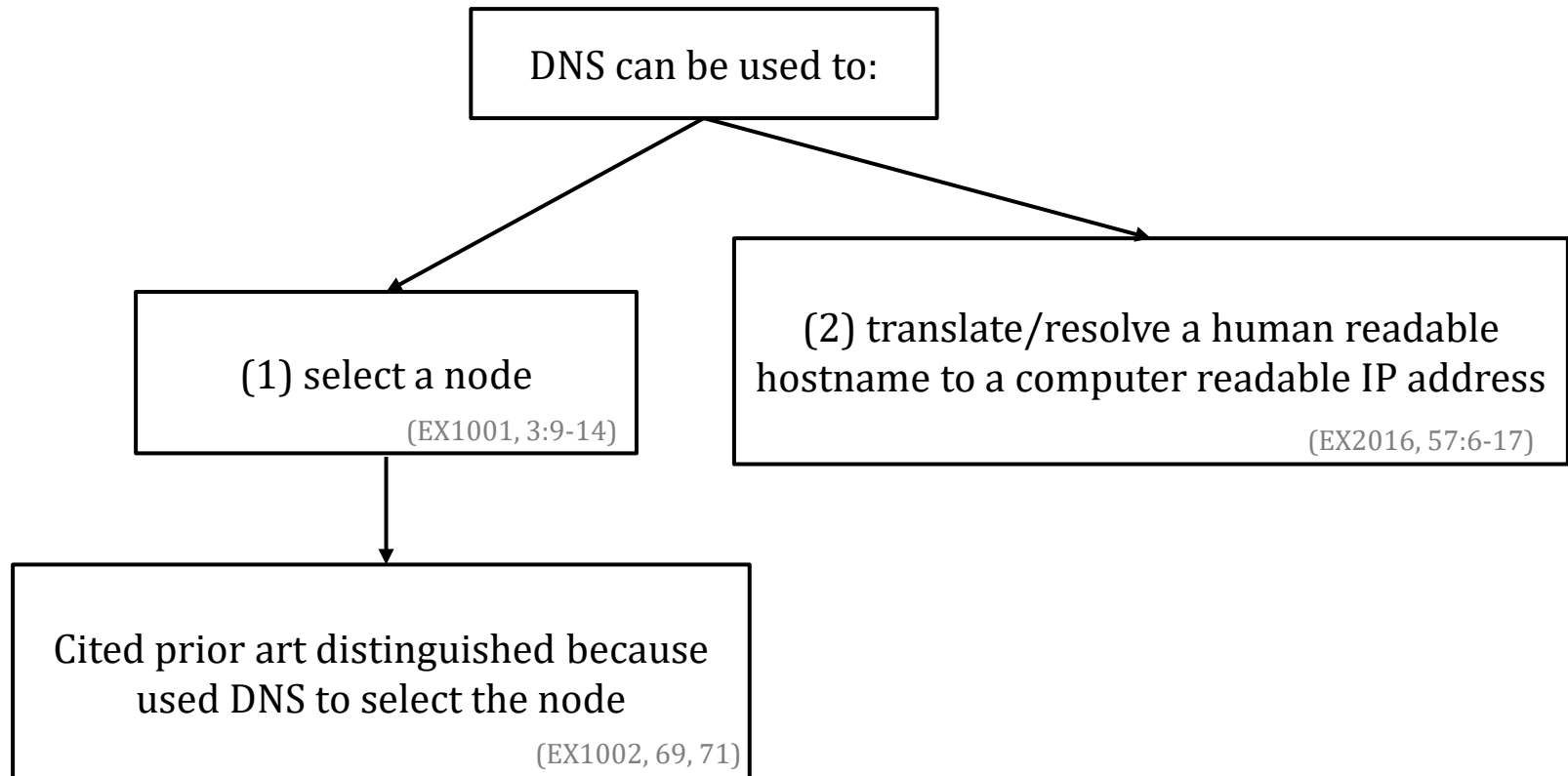
In further support, Google provides webpages with embedded videos. In doing so, Google further generates a modified embedded resource providing a direct link (*e.g.*, a link pointing to specific cache server), as indicated below.

Google provides logic and infrastructure that delivers, by the content delivery network, the modified embedded resource to the origin system. For example, YouTube provides videos requested by users with the link to a specific cache server within Google’s CDN. In one example, the modified embedded resource includes the direct link (*e.g.*, a link pointing) to specific cache server.

EX1019, 7-8

Direct Link to a Node: PO overstates DNS alternative

- Patent Owner contends claims bypass conventional DNS hostname resolution



Direct Link: PO reads out embodiments

- PO reads out hostname embodiments

'322 Patent

The API returns an absolute URL from which the content may be retrieved, and/or a list of hostnames and/or IP addresses that could be used to construct the URLs at the calling application (operation 330). Similarly, the origin server may pass the document to the API, and the API causes the modified embedded resource to be directly embedded and return the modified document to the origin server. Thus, when the client browser displays the Web page, for example, the modified embedded resource (e.g., link) may include an absolute URL for a node (e.g., edge server(s)) within the CDN that can deliver the content. When the user selects the link, the browser will directly contact the node for the content. In the case of a hostname, the embedded resource may include specific reference to the CDN such that name servers 34 of the CDN are contacted directly, bypassing any other DNS processing such that the CDN may perform any form of intelligent traffic management (ITM) or other forms of edge selection and return an IP address of a node within the CDN able to process the request. It may be the case that

Examples of direct links

Avoiding DNS for "edge selection"

EX1001 6:50-7:1

Application Programming Interface (API)

“providing, at a content delivery network, an application programming interface (API)” (claims 2-7, 11-15)

Petitioner	PO	Institution Decision
“‘may include’ the functionality of the engine”	not “collapsed with the underlying engine executing on the server”	API “‘may include’ the functionality of the engine, . . . API need not be separate from the engine executing on the server”
“an interface that allows the client to communicate with an application executing on the server”	“a defined programmatic interface that specifies the inputs a calling system must supply and the output the receiving system will return”	

API may include Compute Engine functionality

- API provides interface between client and application executing on server
- Specification: API may be combined with compute engine; may be “API/compute engine”

'322 Patent

In an alternative implementation, which may also involve an API and calling the same in order to resolve an optimal node to serve a request, the process of contacting the API and constructing the optimal network address is instead handled by the CDN and does not necessarily involve a customer origin server or other devices outside the CDN. While embodiments are discussed herein as involving an API, it should be noted that an API is a convenient and efficient way for disparate systems to uniformly and predictably pass information to the computing engine and for the computing engine to return information. However, other forms of communicating with and passing information to the computing engine are also possible. Moreover, the API may include the functionality of the engine and not necessarily be considered a discrete mechanism from the computing engine functions.

EX. 1001, 7:10-25

As shown, main memory 604 is encoded with an embedded resource application 650-1, which may include functionality described relative the API/compute engine discussed herein and that supports functionality as discussed above and as discussed further below. For example, in one

EX1001, 9:1-5

'322 Patent

present system, the API/compute engine may instead cause the creation of an embedded resource 24B (also referred to as a modified embedded resource) that will direct the client to the New York node. While described herein as an API call (or calls) in conjunction with an associated compute engine, other applications may be used to compute and return the modified embedded resource. In some instances, only an API is referenced but it should be recognized that such reference includes not only the interface but also the associated compute routine, function, and the like for processing inputs to the API and generating a modified embedded resource. In any event, a modified embedded resource is

EX1001, 5:1-12

tagged embedded resource 58A and contact the API/compute node 60 to return an embedded resource (referred to as the “modified embedded resource”) 58B to include in the page returned to the requesting client device 40 (operations 540 and 550).

The API/compute engine will have access to various types of information from various possible sources. For example,

EX1001, 7:51-57

API acts as a Bridge/Interface

- Patent Owner contradicts its pre-institution statements

Pre-Institution

- PO's expert, Dr. Russ, opined that "API" is an "interface" or "bridge" (EX2010, ¶¶50-51)

Post-Institution

- Dr. Russ discards prior opinion (EX2015, ¶105)

My description of an API as a "bridge" was intended to be a general explanation of what an API is as a concept in the field of computer science. EX2010, ¶50. It was not intended to define the metes and bounds of the API limitation as used in the claims of the '322 patent.

EX2015, ¶105

API is not a defined interface with specific inputs and outputs

- PO's expert (Dr. Russ) reads in limitations requiring a specific input-output contract, which lack intrinsic support

make them APIs. The distinction is whether the external caller invokes a specific,

defined interface with a specified parameter-response contract, or whether it

simply sends a message into a system's internal pipeline. Verma's broker receives

EX2015, ¶108

- But the '322 patent provides for flexible inputs and outputs

'322 Patent

The API may obtain information used in determining the address from various sources. To begin, the request from the origin server may include information. Since the client has

Besides client information, including geography, and popularity, the origin server may also pass other information such as policy information, political information, restrictions, etc. For example, the distribution of some content may be geographically limited—e.g., obtaining real-time sporting content may be limited in the geographic area of one or both teams participating in the event. Some content may have geopolitical distribution restrictions. This information may also be stored separately at the CDN or otherwise separately accessible by the CDN and passed to or otherwise used in generating the modified embedded resource. Regard-

EX. 1001, 5:16-27, 5:65-6:8

'322 Patent

The API returns an absolute URL from which the content may be retrieved, and/or a list of hostnames and/or IP addresses that could be used to construct the URLs at the calling application (operation 330). Similarly, the origin

EX. 1001, 6:50-53

POR, 50-51; Pet. Reply, 9-11; PO Sur-Reply, 11-12.

With a Tag

“the content resource including at least one embedded resource with a tag” (claims 1-10)

Petitioner	Patent Owner
“an embedded resource to have a ‘tag’ identifying, associated with, or included in it”	Tag is “is embedded inside or is the URL string itself”

'322 Patent

Embodiments of the present disclosure involve a content delivery method including the operations of receiving a request for a content resource including at least one embedded resource with a tag. Upon identification of the tag, using

EX. 1001, 1:57-60

} Broader disclosure of embedded resource “with a tag”

With a Tag: Not limited to “Including”

- PO reads “may include a tag” embodiment into the claims

'322 Patent

Exemplary embodiment –
uses permissive language
of “may include a tag”

The embedded resource may include a tag indicating that a resource address compute engine 30 should be contacted via a call to an API 26 (operation 220). For instance, the HTML document may include an embedded resource of the form:

`http://{CALLAPI:ADDRONLY}/path/resource.txt, or
{CALLAPI:F'ULLURL,path=$ENV{PATH},
pop=$POP}.`

In both instances, the tag is “\$” which causes the origin server 18 to call the API 26 to retrieve the address (only) or the complete URL. Hence, the presence of the tagged

EX. 1001, 4:32-39

Grounds 1A/1B & 2: Disputed Limitations in *Verma*

Verma Discloses API and Mod ER with Direct Link

Claim 15

[15pre]

[15a] API at CDN

[15b]

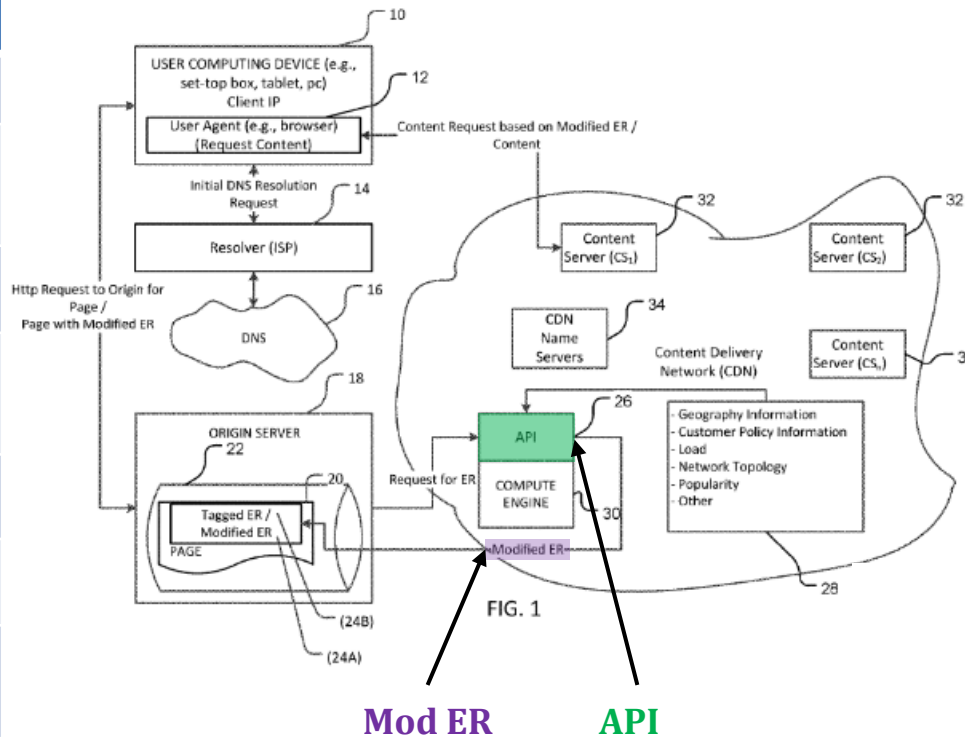
**[15c] Mod ER
with direct link**

[15d]

[15e]

[15f]

'322 Patent, Figure 1 (annotated)



Verma, Figure 1 (annotated)

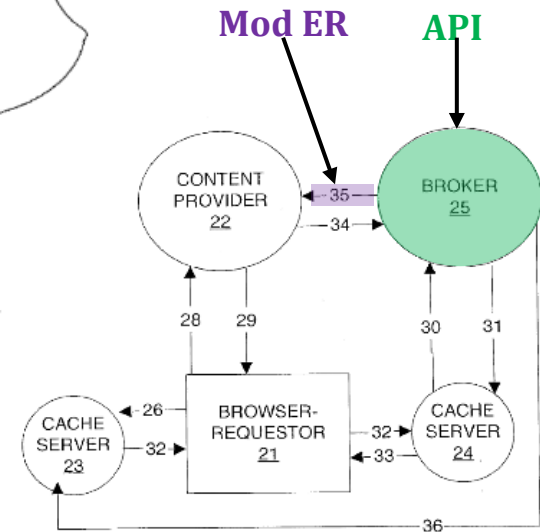


FIG. 1

Pet., 11-12, 23-34; Pet. Reply, 11-13, 24-27.

Verma Direct Link: points to specific server

- Broker/CDN selects a specific node and returns address of that node
- System does not use DNS to select node

Verma

[0013] The present invention enables extending request routing capability to cache servers, which can hold content at a future time period. For example, Akamai's request routing technique is to replace the original URL (e.g. `http://contentURL`) with an "Akamai-ed" URL (e.g. `http://local_cache_server/contentURL`) that redirects to a local cache server in the content delivery network. In accordance with an aspect of Verma, [0013]

[0027] A prepended URL directed to cache server 23 can be redirected to cache server 24 if the cache server 23 determines that the cache server 24 is closer to the client and the content file is available during a time period satisfying the client.

Verma, [0027]

of FIG. 1. If so, a prepended URL with a future availability [T1, T2], redirecting to the cache server 24 of FIG. 1 is returned by the RRFE during step 150 of FIG. 2. When the client receives the prepended URL, it requests the content file from the appropriate cache server when the content is available.

Verma, [0037]

**Node is
already
selected**

```
<Content timing>
<Period start=18:00, 9/10/03, end=19:00>
<a href=http://location1/contentURL> contentURL </a>
</Period>
<Period start=19:00, 9/10/03, end=20:00>
<a href=http://location2/contentURL> contentURL </a>
</Period>
</Content timing>
```

Verma, [0029] (annotated)

Pet., 23-24; Pet. Reply, 11-13.

Verma Direct Link: DNS not used to select the node

Dr. Mowry Deposition Transcript

1 Q Understood. But in that very first
2 instance, that initial mapping of the host name to
3 an IP address can be done using well-known DNS
4 processing techniques, right?

5 MR. FELICIELLO: Objection. Form.

6 A Yes. The translation of a host name to
7 an IP address is -- on the Internet, is typically
8 done through DNS. But, as I said, when the host
9 name has a unique mapping to a specific IP
10 address, it will always return the same mapping.

11 And, in this case, the content delivery network
12 has selected this host name so because it
13 corresponds directly to a particular cache server.
14 So although it goes through DNS it will always
15 return the same IP address.

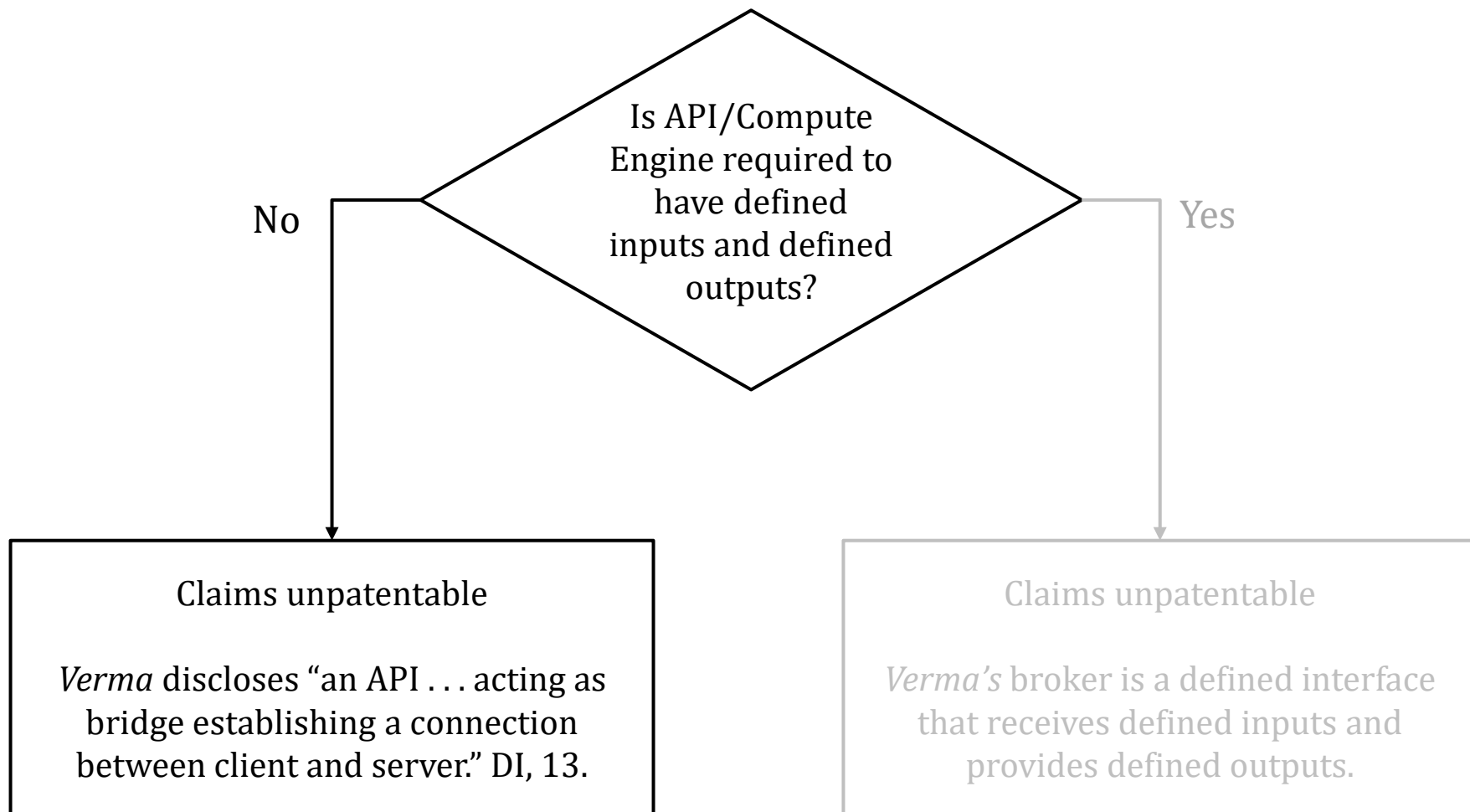
16 And so, it is an absolute URL. It is
17 pointing to a specific cache server.

EX2016, 57:1-17

DNS name
resolution

Node selection:
pointing to
specific server

Verma discloses API



Verma discloses API

Verma

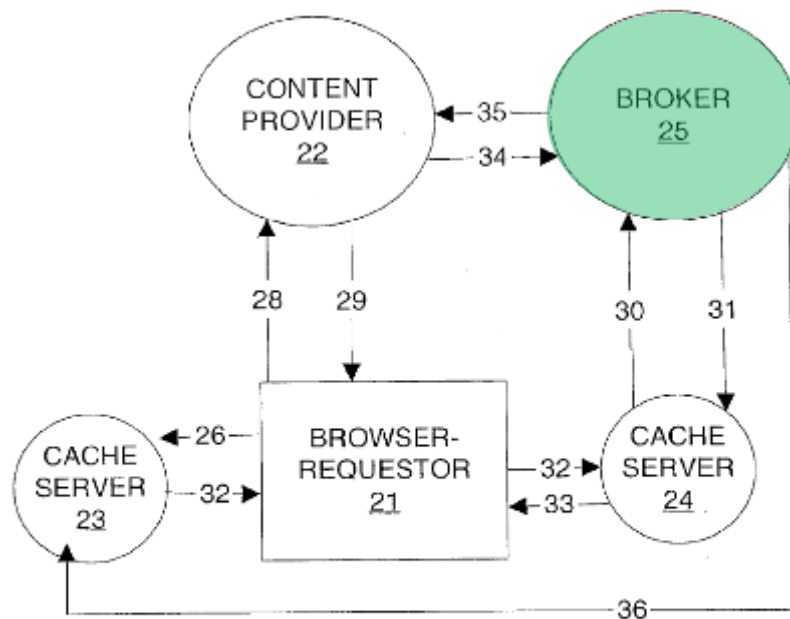


FIG. 1

Verma, FIG. 1 (annotated)

[0025] FIG. 1 depicts a content delivery network in accordance with the present principles. The network includes a client that runs a browser 21. The browser 21 operates to request a content file, for example the movie entitled “Matrix Reloaded,” by sending a URL 28 to a content provider 22, Movies-Inc.Com in this example. Movies-Inc.Com content provider 22 web site has a business relationship with the content delivery network to speed up the delivery of movie files, which are relatively large to its customers, such as client 21. Movies-Inc.Com content provider 22 sends a request 34 for information to a content delivery network broker 25, which manages at least one, and preferably, a set of cache servers, exemplified by cache server 23 and cache server 24. The broker 25 sends availability inquiries 31 and 36 to the cache servers 23 and 24, respectively, to find out whether they currently store the requested movie file. Various movie files are transferred from time to time from the content provider 22 to cache servers 23 and 24 via the broker 25.

Verma, [0025]

Verma Broker discloses API

- Broker is interface/bridge between content providers and the CDN

Verma

[0015] By use of URLs or other comparable mechanisms, the client computer can open web pages and request content files to be transferred or downloaded. The web pages and content files are initially stored at a content server, which runs a web site, but may also be stored in various edge servers in a content delivery network, which comprises a broker, to communicate with content servers. Base on the location and delay

Verma, [0015]

[0022] The content delivery network, which typically comprises at least one edge server and broker(s) to communicate with the content server(s), can schedule the distribution of content files to edge servers and return a corresponding URL with availability time period information to content server through a broker.

Verma, [0022]

[0026] The broker 25 provides content current and/or future availability information to Movies-Inc.Com content provider 22 where the content file URLs are prepended (Aka-mai-ed) for redirecting to cache servers. Additionally, information on the time periods when the content files will be available at corresponding cache servers is also added. Upon receiving a content request from the client 21, the Movies-Inc.Com content provider 22 can also dynamically request the broker 25 for a content file delivery to a location close to the client. Based on the capacity of network links and cache servers, a content delivery schedule can be made by the content delivery network and the content file availability can be returned to the Movies-Inc.Com content provider 22 through the broker 25.

Verma, [0026]

Verma Broker discloses API

Dr. Mowry Declaration

87. **Application Programming Interface:** First, in my opinion, Verma's "broker" is an application programming interface (API) because it "communicate[s] with the content server(s)" and provides an interface between the content server and CDN. *Id.*, [0015], [0022], [0025]-[0026] (explaining "content provider 22 can also dynamically request the broker 25 for a content file delivery to a location close to the client"). For example, "content provider 22 sends a request 34 for information to a content delivery network broker 25," and "broker 25 provides" information back to "content provider 22." *Id.*, [0025]-[0026], FIG. 1. Verma's "broker" is an interface allowing the content provider and content delivery network to interact and exchange data, which a POSITA would have understood is an API (application programming interface). *Id.*, [0015], [0022], [0025], FIG. 1. A POSITA would have also

interface). *Id.*, [0015], [0022], [0025], FIG. 1. A POSITA would have also understood Verma's "broker" is the claimed "application programming interface (API)" because it performs the same logical functions as the API in the '322 patent. For example, the '322 patent explains "[w]hen the origin server receives the request, . . . the origin server calls an API" to "communicate with the compute engine" in the content delivery network. EX-1001, 4:49-52, 7:16-25 ("an API is a convenient and efficient way for disparate systems to uniformly and predictably pass information to the computing engine and for the computing engine to return information"; "the API may include the functionality of the engine"). As the API in the '322 patent provides an interface for communication between a content provider and content delivery network, so too does the "broker" in Verma. EX-1005, [0015], [0022], [0025]-[0026], FIG. 1.

EX1003, ¶87

Verma Broker: Bridge between client and server

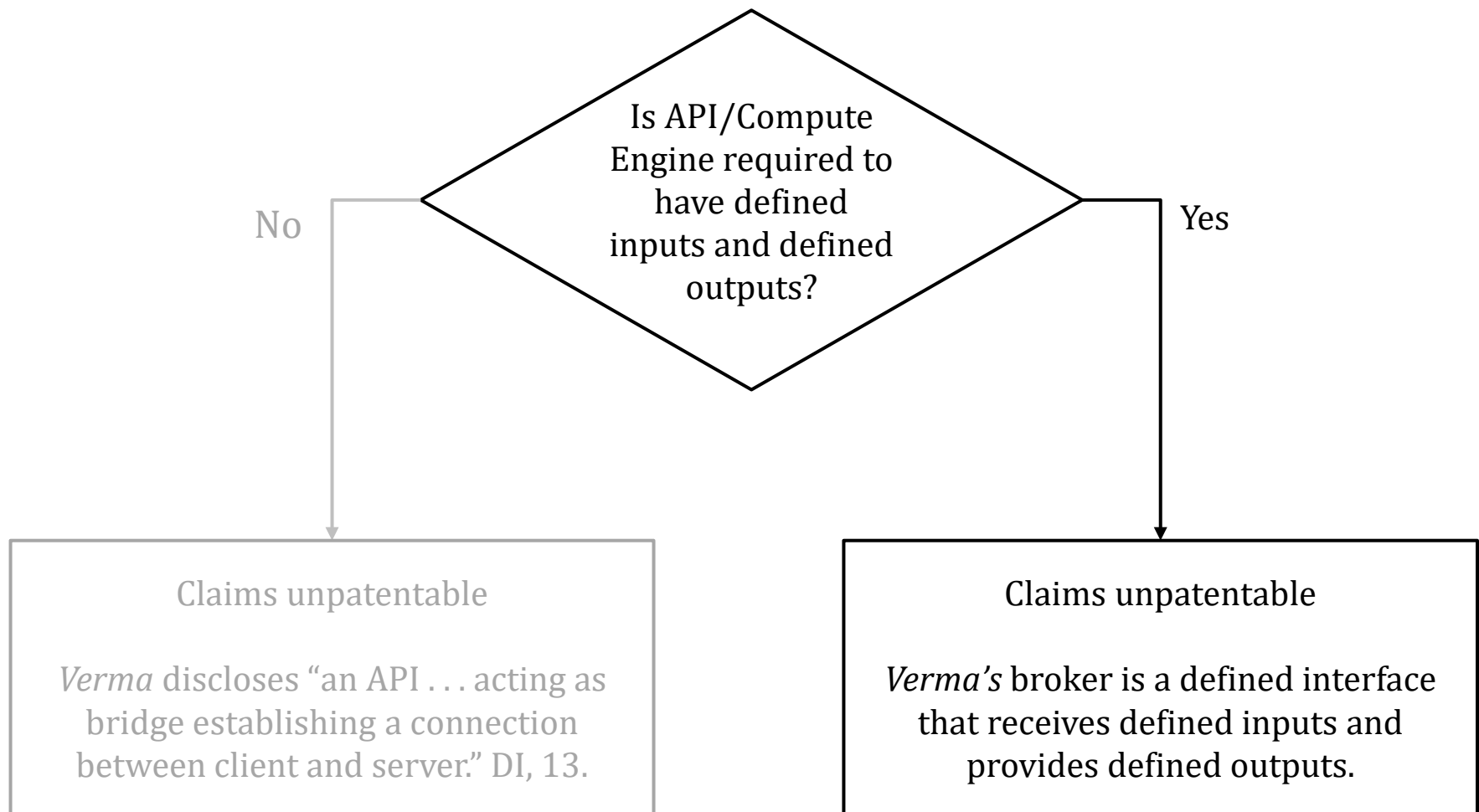
PO's Expert Dr. Russ (As Noted in DI)

We are persuaded by Petitioner's assertions for limitations [15pre]–[15a]. Dr. Mowry attests that Verna's client browser sends a URL 28 to a content provider 22, which is then forwarded to broker 25, which selects the appropriate cache server to serve the content and returns a re-written URL to the content server that received the client request. Ex. 1003 ¶ 67 (citing Ex. 1005 ¶¶ 12, 26, 27, 37). On the current record, this function of Verna's broker fulfills Dr. Russ's characterization of an API as acting as bridge establishing a connection between client and server. Ex. 2010 ¶ 50.

Verma also meets Dr. Russ's characterization of an API as "an interface that allows the client to communicate with an application executing on the server." Ex. 2010 ¶ 50. Dr. Russ describes Verma's broker as sending availability inquiries from the content provider (client) to servers 23 and 24 to "find out whether they currently store the requested movie file," and provide that information back to the content provider (client). *Id.* ¶ 53. On the current record, this function of Verna's broker fulfills Dr. Russ's characterization of an API acting as an interface that allows the client to communicate with an application executing on the server (that which provides availability information to the broker).

DI, 13-14

Verma discloses API even under PO construction



Verma Broker: defined inputs and outputs

- Broker provides interface between CDN and content provider
- Broker accepts defined parameters from content server (defined input)
- Broker transmits prepended URL to content server (defined output)

Verma

[0022] The content delivery network, which typically comprises at least one edge server and broker(s) to communicate with the content server(s), can schedule the distribution of content files to edge servers and return a corresponding URL with availability time period information to content server through a broker.

Verma, [0022]

[0038] If the content file is not immediately available and there is no scheduled availability on any of the cache server close to the client, the content server 22 will negotiate with the content delivery network (CDN) through the broker 25 in order to determine the start time, end time, and the cache server where the content file will next be available during step 160 of FIG. 2 and return a URL redirecting to such cache server with the time period information during step 170. The

Verma, [0038]

municate with content servers. Base on the location and delay tolerance of the request from the client computer, a content server can provide a redirected URL comprising a time period of availability of the corresponding content. The delay tolerance can be, for example, “no later than 6 PM” or “best effort,” for example.

Verma, [0015]

available at corresponding cache servers is also added. Upon receiving a content request from the client 21, the Movies-Inc.Com content provider 22 can also dynamically request the broker 25 for a content file delivery to a location close to the client. Based on the capacity of network links and cache

Verma, [0025]

Verma Broker: matches claim specificity

- Claims require “passing information” to API (claim 2)
- Or “receiving” information at API (claims 11, 15)

'322 Patent, Claims 2, 15

15. A method comprising:

providing, at a content delivery network, an application programming interface (API);

at the API, receiving, from an origin system that is separate from the content delivery network, an embedded resource and at least one parameter associated with the embedded resource;

generating a modified embedded resource providing a direct link to a node within the content delivery network for obtaining content associated with the embedded resource;

delivering, by the content delivery network, the modified embedded resource to the origin system;

receiving, at the node, a request from a client device to obtain the content; and

providing the content to the client device.

EX1001, 12:5-20

2. The content delivery method of claim 1 wherein the request is received over an http connection and the at least one delivery parameter includes an IP address of the client device, and further comprising:

passing information based on the client IP address to an application programming interface (API) of the content delivery network;

in association with the API, obtaining a parameter associated with the embedded resource;

identifying, based on the IP address and the parameter, location information for the node in the content delivery network that can serve the content associated with the embedded resource; and

generating the modified embedded resource to include the direct link to the node.

EX1001, 10:19-33

Grounds 3-5: Disputed Limitations in *Verma + Lewin*

Verma + Lewin Disputes

1. Whether *Verma* discloses “direct link to a node” (claims 1-10)
2. Whether *Verma* discloses “application programming interface” (claims 2-7)
3. Whether *Verma + Lewin* renders obvious “tag” limitations
 - Whether Lewin discloses tags (claims 1-10)
 - Whether Lewin discloses “upon identification of tag” (claims 1-10)
 - MTC for tag limitations (claims 1-10)
 - MTC *Verma/Lewin* for content resource (claims 1-10, 13)

Verma/Lewin Renders Obvious Tag Limitations

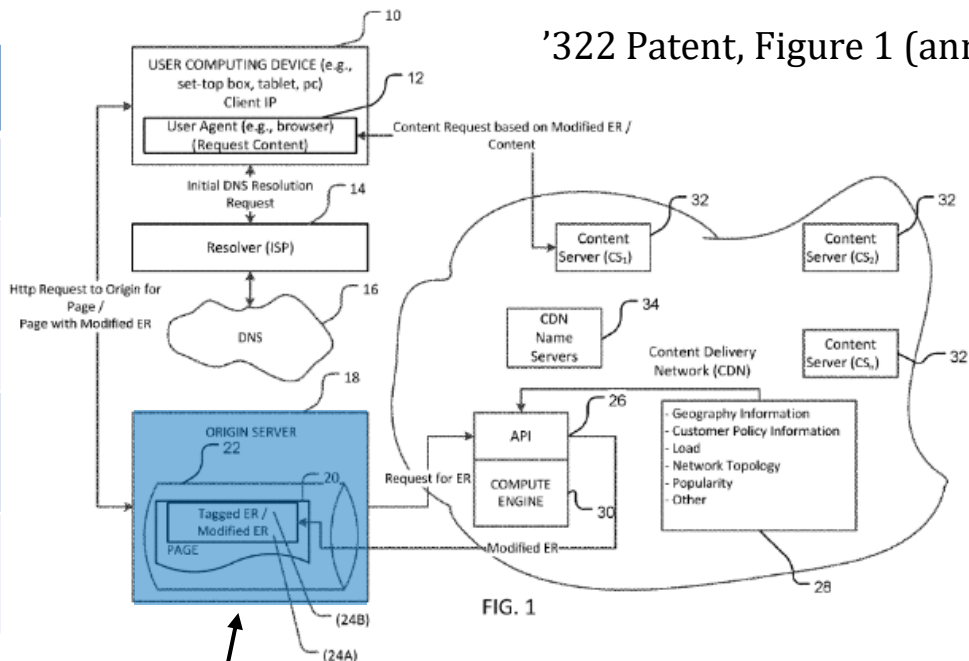
Claim 1

[1pre]

[1a] Content resource with ER and tag

[1b] Identifying tag at origin system

[1c] Delivering Mod ER with direct link



Identifying tags
in content
resource

1. USER REQUEST FOR MY.XYZ.COM IS DIRECTED TO OPTIMAL AKAMAI SERVER.

2. EdgeSuite PARSES THE TEMPLATE, WHICH MAY BE CACHED ALREADY, LOOKING FOR TAGS AND INSTRUCTIONS TO ASSEMBLE THE PAGE AND SEND INDIVIDUALIZED CONTENT TO USER.

Lewin, Figure 9
(annotated)

3. IF NECESSARY, EdgeSuite CALLS XYZ.COM OVER PERSISTENT TCP CONNECTION TO OBTAIN NEW OR UNCACHED HTML FRAGMENTS.

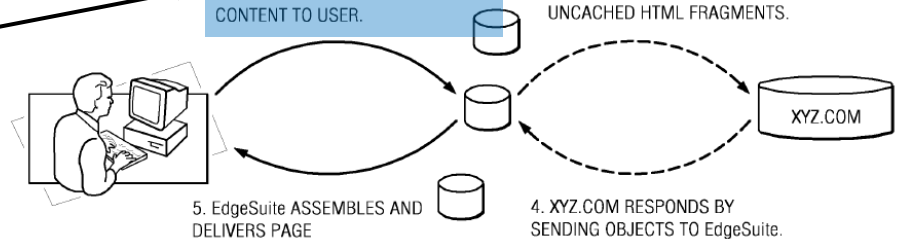


FIG. 9

Verma discloses requesting content

- “receiving, by an origin system, a request from a client device for a content resource, the content resource including at least one embedded resource with a tag”

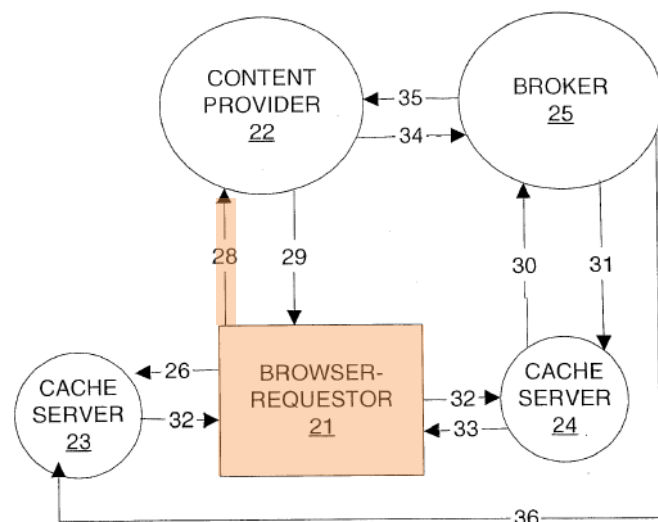


FIG. 1

Verma, Fig. 1 (annotated)

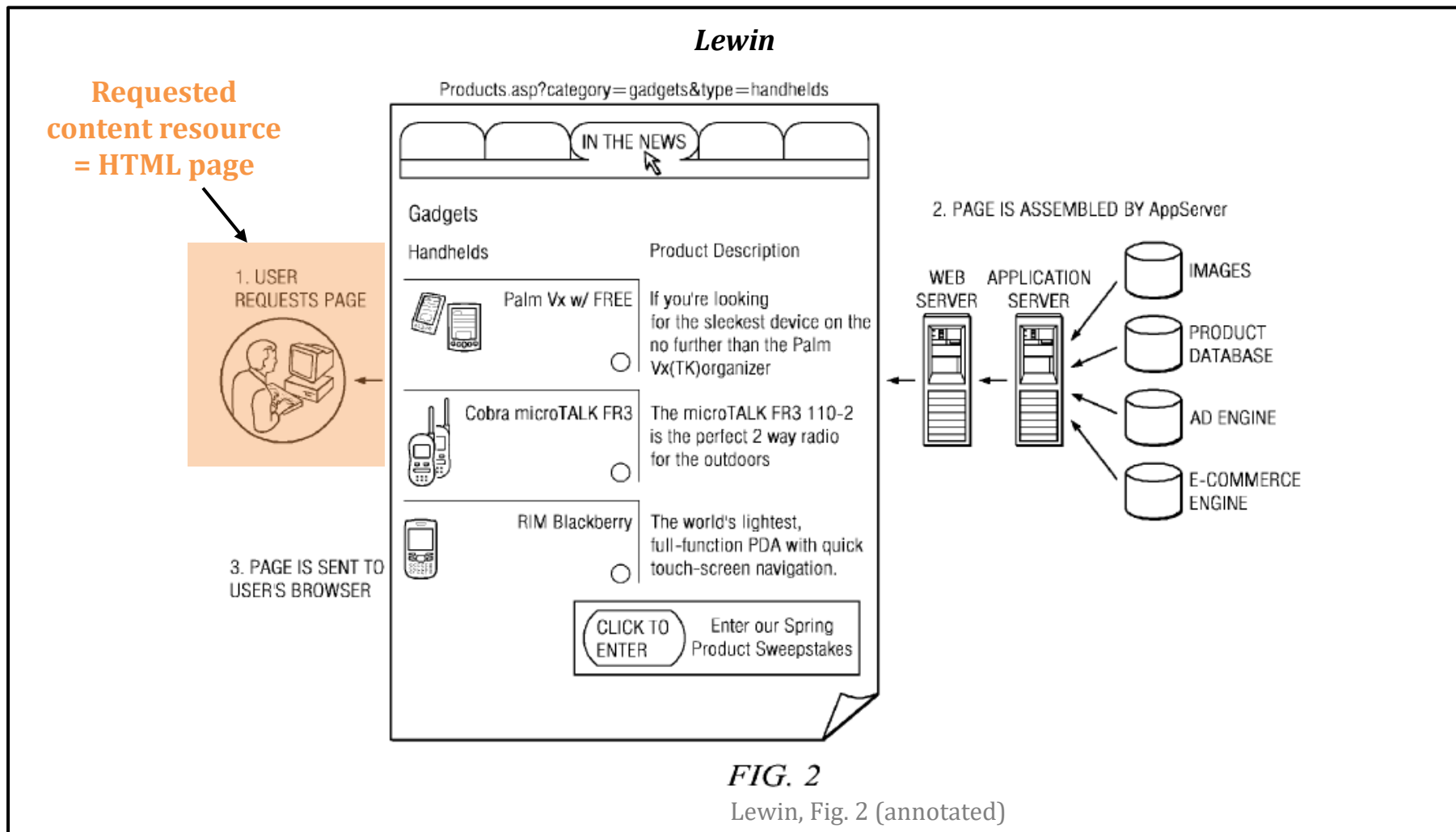
Verma

[0025] FIG. 1 depicts a content delivery network in accordance with the present principles. The network includes a client that runs a browser 21. The browser 21 operates to request a content file, for example the movie entitled “Matrix Reloaded,” by sending a URL 28 to a content provider 22, Movies-Inc.Com in this example. Movies-Inc.Com content provider 22 web site has a business relationship with the content delivery network to speed up the delivery of movie files, which are relatively large to its customers, such as client 21. Movies-Inc.Com content provider 22 sends a request 34 for information to a content delivery network broker 25, which manages at least one, and preferably, a set of cache servers, exemplified by cache server 23 and cache server 24. The broker 25 sends availability inquiries 31 and 36 to the cache servers 23 and 24, respectively, to find out whether they currently store the requested movie file. Various movie files are transferred from time to time from the content provider 22 to cache servers 23 and 24 via the broker 25.

Verma, [0025]

Lewin discloses requesting an HTML page

- “receiving, by an origin system, a request from a client device for a content resource, the content resource including at least one embedded resource with a tag”



MTC: Obvious to request an HTML page in *Verma*

- Advantageous to request HTML page rather than typing in lengthy URL
- Only disputed issue for claim 13 *Verma/Lewin* mapping

Dr. Mowry's Declaration

115. Verma teaches its client uses a “web browser such as Internet Explorer, or Netscape.” *Id.*, [0014], [0025]. A POSITA would have understood typing lengthy URLs into a browser for each content item sought would be inconvenient. A browser that would instead interact with embedded options in the webpage, such as a plurality of movie options, would have advantageously allowed the client to survey and select numerous movies quickly without needing to provide a lengthy URL into the browser to access each movie. Indeed, there was a well-known desire in the art to reduce the number of lengthy URLs that users had to type into browsers. *See, e.g.*, EX-1016 (Rossmann), 2:24-26 (explaining “it is quite laborious to enter a long

alphanumeric Universal Resource Locator (URL) to access a specified service using the phone keypad”). When a user is able to click on an embedded link, it would have greatly increased the efficiency of browsing. A POSITA would have known this was how users of browsers ordinarily interacted with the Internet by December 31, 2014, including for example, on video websites, where users would have scrolled through a plurality of embedded videos and selected a video by clicking on an embedded link, without needing to type a lengthy URL for each individual video. A POSITA would have known that this desired effect of quick and efficient access stemmed from embedded links in web pages that users accessed via their browsers.

EX1003, ¶¶113-115; *see also* EX1003, ¶¶199-204, 208-211

Lewin discloses an Embedded Resource with a Tag

- “receiving, by an origin system, a request from a client device for a content resource, the content resource including at least one embedded resource with a tag”

```
<html>
<body>
<!--personalized greeting (1)-->
Hello ${HTTP_COOKIE{"username"}}

<!--targeted ad (2)-->
<esi:choose>
<esi:when test="S(GEO{country_code})='US' ">
  <esi:include src=us_ad.html/>
</esi:when>
<esi:when test="S(GEO{country_code})='Canada' ">
  <esi:include src=canada_ad.html/>
</esi:when>
<esi:otherwise>
  <esi:include src=generic_ad.html/>
</esi:otherwise>
</esi:choose>

<!--Static navigation bar (3)-->
<a href=...> <a href=...> <a href=...> <a href=...>

<!--Personalized recommendations (4)-->
<esi:include src=recommendations.html/>

<!--Static links, copyright,etc (5)-->
<a href=...> <a href=...> <a href=...> <a href=...>
Copyright 2001, etc.
</body>
</html>
```

Lewin

← Content resource (HTML page)

← Exemplary tag

← Exemplary embedded resource

necessary (not shown). At step (2), the edge server parses the HTML page looking for tags that specify dynamic assembly instructions including, without limitation, URLs for HTML chunks to be incorporated in the final HTML page. The DCA

Lewin, [0049]

FIG. 6 Lewin, Fig. 6 (annotated)

Lewin discloses Identifying Tags

- “upon identification of the tag, using at least one delivery parameter to obtain, by the origin system, a modified embedded resource from a content delivery network that is separate from the origin system”

Lewin

necessary (not shown). At step (2), the edge server parses the HTML page looking for tags that specify dynamic assembly instructions including, without limitation, URLs for HTML chunks to be incorporated in the final HTML page. The DCA tags are preferably specified according to ESI, or some other server side scripting language, as has been described generally above.

Lewin, [0049]

HTML page is then sent to the requesting end user. The HTML page may contain URLs or other CDN-modified resource locators to other embedded page objects such as .gif, .jpg, or the media-rich content, which is then requested from the CDN. The CDN delivers this content to complete the

Lewin, [0050]

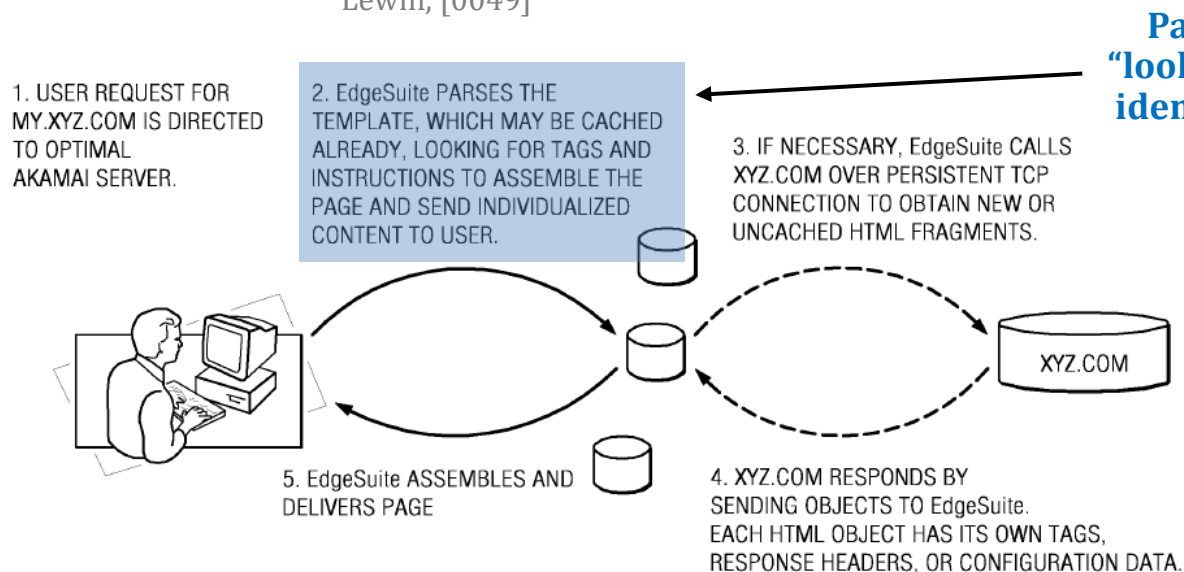


FIG. 9

Lewin, Fig. 9 (annotated)

MTC: Obvious to have HTML page include embedded resource with a tag

- Advantageous to identify tags

Dr. Mowry's Declaration

212. Content Resource Including an Embedded Resource with a Tag: It

would have been obvious for Verma's "URL 28" to be embedded in an HTML page and specified by a "tag[]," for Lewin teaches a "central" webpage development feature "is the ability to fetch and include files that comprise a web page," and "tag[s]" "may be used for this purpose." §§ X.A.3, X.C (Grounds 1A/1B, [15b], [12]), VIII.E; EX-1005, [0025]; EX-1007, [0028], [0030], [0049].

EX1003, ¶212

instructions including . . . , URLs." *Id.*, [0049]. The POSITA would have understood Lewin's tags advantageously allow for specification and identification of URLs, and would have been motivated to incorporate those tags into Verma's system to provide a standard, uniform, and more efficient and uniform way to identify URLs. *Id.*, [0028], [0049].

EX1003, ¶214; *see also* EX1003, ¶¶ 213, 215-217

MTC: *Verma's* tags reinforce expectation of success

- *Verma* suggests tags with *modified* embedded resource
- *Lewin's* tags are with embedded resource, as claimed in '322 patent

Petition's Mapping of *Verma*

Similarly, *Verma* suggests URLs with tags in its modified embedded resource, as discussed for [15b] and claim 12. EX-1005, [0029] (exemplary absolute URL with “<a href= . .” tags); §§ VI.A.3, VI.C, VII.A.2, VII.C, VIII.A.2.a (Grounds 1A/1B/2, [15b], [12]), II.D; EX-1003 ¶223. And as shown below:

```
<Content timing>
<Period start=18:00, 9/10/03, end=19:00>
<a href=http://location1/contentURL> contentURL </a>
</Period>
<Period start=19:00, 9/10/03, end=20:00>
<a href=http://location2/contentURL> contentURL </a>
</Period>
</Content timing>
```

Tag with modified embedded resource

EX-1005, [0029] (tags in blue). Markup language tags were known to identify URLs in webpage design, as discussed, and it would have required no more than the application of conventional technology to designate an original URL (embedded resource) with a tag in a webpage (resource). §§ II.D, VIII.A.2 (Ground 3, [1a]); EX-
Verma, [0029] (annotated); EX1003, ¶223

Lewin

```
<html>
<body>
<!--personalized greeting (1)-->
Hello ${HTTP_COOKIE{"username"}}

<!--targeted ad (2)-->
<esi:choose>
<esi:when test="${GEO{country_code}}='US' ">
  <esi:include src=us_ad.html/>
</esi:when>
<esi:when test="${GEO{country_code}}='Canada' ">
  <esi:include src=canada_ad.html/>
</esi:when>
<esi:otherwise>
  <esi:include src=generic_ad.html/>
</esi:otherwise>
</esi:choose>

<!--Static navigation bar (3)-->
<a href=...> <a href=...> <a href=...> <a href=...>

<!--Personalized recommendations (4)-->
<esi:include src=recommendations.html/>

<!--Static links, copyright, etc (5)-->
<a href=...> <a href=...> <a href=...> <a href=...>
Copyright 2001, etc.
</body>
</html>
```

Tag with embedded resource

“href” and server-side include tags in same HTML page

FIG. 6

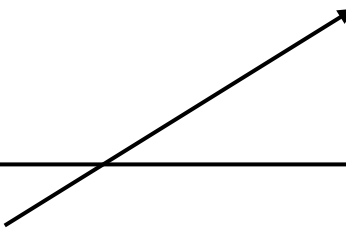
Lewin, FIG. 6 (annotated)

Identifying Tags: PO relies on unclaimed feature

Patent Owner Sur-Reply

See Reply, 22, 24; EX1007, ¶50. Lewin's tags merely trigger retrieval of the HTML chunks, which may happen to include resource locators that have been modified by the CDN. EX1007, ¶50. Identification of these tags does not trigger a modification of these resource locators.

not claim
requirement



MTC: No bodily incorporation

- *Verma* teaches claimed processing direction
- *Lewin's* teachings apply broadly to any “server side include” language

Verma's Existing Processing Direction

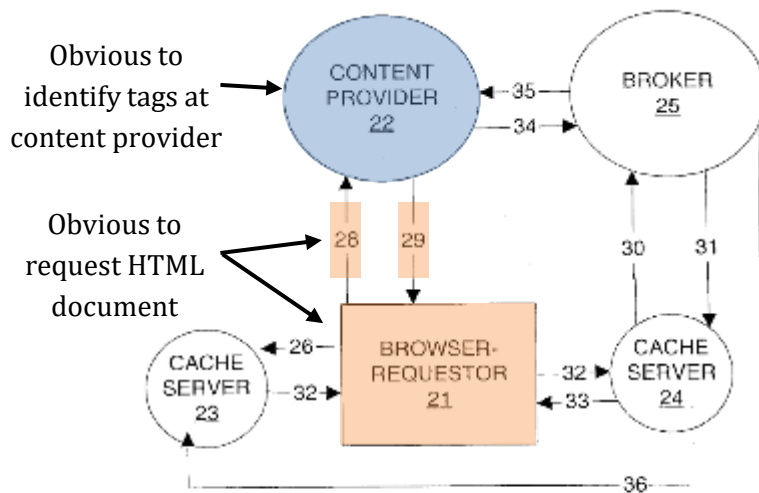


FIG. 1

Verma, FIG. 1 (annotated)

Lewin

[0028] The dynamic content assembly mechanism of this disclosure leverages any server side scripting language or other server-based functionality. In a preferred embodiment, the functionality is a variant of server side include processing that is sometimes referred to as an “edge side include” to emphasize that the processing is carried out on an edge server.

Lewin, [0028]

chunks to be incorporated in the final HTML page. The DCA tags are preferably specified according to ESI, or some other server side scripting language, as has been described generally above.

Lewin, [0049]

MTC: No bodily incorporation

- *Lewin's* teachings apply to the conventional processing direction disclosed in *Verma*

Dr. Mowry's Declaration

“When a user accesses the site and searches for, say, Handhelds, that request is sent to the application server.” *Id.* “The application server performs a database query and assembles the page based on the return values and other common page components, such as navigation menu, logos and advertising.” *Id.* “The user then receives the assembled page containing product images, product descriptions, and advertising.”

This process conventionally “require[es] a full round-trip from browser to origin server and back.” *Id.*

206. In my opinion, a POSITA would have understood that Lewin seeks to improve upon this conventional system. *E.g.*, [0008], [0009]. However, a POSITA would have understood Lewin describes conventional web content access framework, i.e., the concept of requesting an HTML page, parsing the HTML for pages| used to identify embedded content, and then fetching or modifying the embedded content before serving the page to the client user. EX-1007, [0005], [0006], [0007], [0008], [0013], [0028], [0029], [0030], [0049], [0050]; *see also* EX-

EX1003, ¶¶ 205-207