Doc Code: PA..
Document Description: Power of Attorney

PTO/AIA/82A (07-13)
Approved for use through 03/31/2021. OMB 0651-0035
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

TRANSMITTAL FOR POWER OF ATTORNEY TO ONE OR MORE REGISTERED PRACTITIONERS

NOTE: This form is to be submitted with the Power of Attorney by Applicant form (PTO/AIA/82B) to identify the application to which the Power of Attorney is directed, in accordance with 37 CFR 1.5, unless the application number and filing date are identified in the Power of Attorney by Applicant form. If neither form PTO/AIA/82A nor form PTO/AIA82B identifies the application to which the Power of Attorney is directed, the Power of Attorney will not be recognized in the application.

Application Number	11932553
Filing Date	10/31/2007
First Named Inventor	Kembel
Title	DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES
Art Unit	2478
Examiner Name	LIN
Attorney Docket Number	0416-001B

SIGNATURE of Applicant or Patent Practitioner

Signature	/Peter G Korytnyk/	Date (Optional)	May 12, 2022
Name	Peter G Korytnyk	Registration Number	43400
Title (if Applicant is a juristic entity)			
Applicant Name (if Applicant is a juristic entity)			

NOTE: This form must be signed in accordance with 37 CFR 1.33. See 37 CFR 1.4(d) for signature requirements and certifications. If more than one applicant, use multiple forms.

☐ *Total of _____ forms are submitted.

This collection of information is required by 37 CFR 1.131, 1.32, and 1.33. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and 1.14. This collection is estimated to take 3 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

SAMSUNG 1002

GENERAL POWER OF ATTORNEY BY APPLICANT AND/OR ASSIGNEE
FOR UNITED STATES PATENT

U.S. DEPARTMENT OF COMMERCE
Patent and Trademark Office

I hereby revoke all previous powers of attorney given in the application identified in the attached transmittal letter (form PTO/AIA/82A or equivalent).

I hereby appoint Practitioner(s) associated with the following Customer Number for **PK Patent Law, LLC:**

41200

as my attorney(s) or agent(s), and to transact all business in the United States Patent and Trademark Office in connection with for the application referenced in the attached transmittal letter (form PTO/AIA/82A or equivalent). Please recognize or change the **correspondence address** for the application identified in the attached transmittal letter to the address associated with the above-mentioned **Customer Number**.

I am the:

Inventor or Joint Inventor.

Legal Representative of a Deceased or Legally Incapacitated Inventor

Assignee or Person to Whom the Inventor is Under an Obligation to Assign and am authorized to act on behalf of all assignees.

Person Who Otherwise Shows Sufficient Proprietary Interest (e.g., a petition under 37 CFR 1.46(b)(2) was granted in the application or is concurrently being filed with this document)

Assignee Name & Address:
DoDots Licensing Solutions, LLC
1701 Directors Blvd., Suite 300
Austin, TX 78744-1044

Signature of Applicant and/or Assignee:

Signature: _____ Date: May 6, 2022

Name (printed): David White

Telephone: (949) 939-0775

Title and Company: Sole Member, DoDots Licensing Solutions, LLC

PTO/AIA/96 (08-12)
Approved for use through 01/31/2013. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

STATEMENT UNDER 37 CFR 3.73(c)

Applicant/Patent Owner: DoDots Licensing Solutions, LLC

Application No./Patent No.: 8510407 Filed/Issue Date: 08-13-2013

Titled: DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES

DoDots Licensing Solutions, LLC, a limited liability company

(Name of Assignee) (Type of Assignee, e.g., corporation, partnership, university, government agency, etc.)

states that, for the patent application/patent identified above, it is (choose **one** of options 1, 2, 3 or 4 below):

1. ☑ The assignee of the entire right, title, and interest.

2. ☐ An assignee of less than the entire right, title, and interest (check applicable box):

   ☐ The extent (by percentage) of its ownership interest is _____%. Additional Statement(s) by the owners holding the balance of the interest must be submitted to account for 100% of the ownership interest.

   ☐ There are unspecified percentages of ownership. The other parties, including inventors, who together own the entire right, title and interest are:

   Additional Statement(s) by the owner(s) holding the balance of the interest must be submitted to account for the entire right, title, and interest.

3. ☐ The assignee of an undivided interest in the entirety (a complete assignment from one of the joint inventors was made). The other parties, including inventors, who together own the entire right, title, and interest are:

   Additional Statement(s) by the owner(s) holding the balance of the interest must be submitted to account for the entire right, title, and interest.

4. ☐ The recipient, via a court proceeding or the like (*e.g.*, bankruptcy, probate), of an undivided interest in the entirety (a complete transfer of ownership interest was made). The certified document(s) showing the transfer is attached.

The interest identified in option 1, 2 or 3 above (not option 4) is evidenced by either (choose **one** of options A or B below):

A. ☐ An assignment from the inventor(s) of the patent application/patent identified above. The assignment was recorded in the United States Patent and Trademark Office at Reel _____, Frame _____, or for which a copy thereof is attached.

B. ☑ A chain of title from the inventor(s), of the patent application/patent identified above, to the current assignee as follows:

1. From: Inventors To: DODOTS, INC.

   The document was recorded in the United States Patent and Trademark Office at
   Reel 039760, Frame 0513, or for which a copy thereof is attached.

2. From: DODOTS, INC. To: SHERWOOD PARTNERS, INC.

   The document was recorded in the United States Patent and Trademark Office at
   Reel 039760, Frame 0850, or for which a copy thereof is attached.

This collection of information is required by 37 CFR 3.73(b). The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

PTO/AIA/96 (08-12)
Approved for use through 01/31/2013. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

STATEMENT UNDER 37 CFR 3.73(c)

3. From: SHERWOOD PARTNERS, INC. To: INNOVATION MANAGEMENT SCIENCES

The document was recorded in the United States Patent and Trademark Office at
Reel 039760, Frame 0935, or for which a copy thereof is attached.

4. From: INNOVATION MANAGEMENT SCIENCES To: MAINSTREAM SCIENTIFIC, LLC

The document was recorded in the United States Patent and Trademark Office at
Reel 039760, Frame 0988, or for which a copy thereof is attached.

5. From: MAINSTREAM SCIENTIFIC, LLC To: STRATEGIC INTELLECTUAL SOLUTIONS, LLC

The document was recorded in the United States Patent and Trademark Office at
Reel 044474, Frame 0829, or for which a copy thereof is attached.

6. From: STRATEGIC INTELLECTUAL SOLUTIONS, LLC To: DODOTS LICENSING SOLUTIONS, LLC

The document was recorded in the United States Patent and Trademark Office at
Reel 044474, Frame 0836, or for which a copy thereof is attached.

☐ Additional documents in the chain of title are listed on a supplemental sheet(s).

☐ As required by 37 CFR 3.73(c)(1)(i), the documentary evidence of the chain of title from the original owner to the assignee was, or concurrently is being, submitted for recordation pursuant to 37 CFR 3.11.

[NOTE: A separate copy (i.e., a true copy of the original assignment document(s)) must be submitted to Assignment Division in accordance with 37 CFR Part 3, to record the assignment in the records of the USPTO. See MPEP 302.08]

The undersigned (whose title is supplied below) is authorized to act on behalf of the assignee.

/Peter G. Korytnyk/	June 2, 2022
Signature	Date
Peter G. Korytnyk	43400
Printed or Typed Name	Title or Registration Number

[Page 2 of 2]

Privacy Act Statement

The **Privacy Act of 1974 (P.L. 93-579)** requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (*i.e.*, GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Electronic Acknowledgement Receipt	
EFS ID:	45859889
Application Number:	11932553
International Application Number:	
Confirmation Number:	2604
Title of Invention:	DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES
First Named Inventor/Applicant Name:	John Albert Kembel
Customer Number:	909
Filer:	Peter George Korytnyk
Filer Authorized By:	
Attorney Docket Number:	043311-0381668
Receipt Date:	02-JUN-2022
Filing Date:	31-OCT-2007
Time Stamp:	14:11:40
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Power of Attorney	POA0416_001B.pdf	1966668 8d425d9f076e8d843512 06c38e4f3ce2bfc4375e	no	2
Warnings:					

Information:					
2	Assignee showing of ownership per 37 CFR 3.73	POA373STMT0416_001B.pdf	125111 eb796b8d10629c71be6a836824a1dcc1e00ce72c	no	3
Warnings:					
Information:					
			Total Files Size (in bytes):	2091779	

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

AO 120 (Rev. 08/10)

TO:	**Mail Stop 8** **Director of the U.S. Patent and Trademark Office** **P.O. Box 1450** **Alexandria, VA 22313-1450**	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court Western District of Texas - Waco Division on the following

☐ Trademarks or ☒ Patents. (☐ the patent action involves 35 U.S.C. § 292.):

DOCKET NO.	DATE FILED	U.S. DISTRICT COURT
6:22-cv-00535	May 24, 2022	Western District of Texas - Waco Division

PLAINTIFF	DEFENDANT
DoDots Licensing Solutions LLC	Samsung Electronics Co., LTD., Samsung Electronics America, Inc., Best Buy Stores, L.P., BestBuy.com, LLC, and Best Buy Texas.com, LLC

	PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1	9,369,545	June 14, 2016	DoDots Licensing Solutions LLC
2	8,020,083	September 13, 2011	DoDots Licensing Solutions LLC
3	8,510,407	August 13, 2013	DoDots Licensing Solutions LLC
4			
5			

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading	
PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT

CLERK	(BY) DEPUTY CLERK	DATE

Copy 1—Upon initiation of action, mail this copy to Director **Copy 3—Upon termination of action, mail this copy to Director**
Copy 2—Upon filing document adding patent(s), mail this copy to Director **Copy 4—Case file copy**

AO 120 (Rev. 08/10)

TO:	**Mail Stop 8** **Director of the U.S. Patent and Trademark Office** **P.O. Box 1450** **Alexandria, VA 22313-1450**	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court Western District of Texas - Waco Division on the following

☐ Trademarks or ☒ Patents. (☐ the patent action involves 35 U.S.C. § 292.):

DOCKET NO.	DATE FILED	U.S. DISTRICT COURT
6:22-cv-00533	May 24, 2022	Western District of Texas - Waco Division

PLAINTIFF	DEFENDANT
DoDots Licensing Solutions LLC	Apple Inc., Best Buy Stores, L.P., BestBuy.com, LLC, and Best Buy Texas.com, LLC

	PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1	9369545	June 15, 2016	DoDots Licensing Solutions LLC
2	8020083	September 13, 2011	DoDots Licensing Solutions LLC
3	8510407	August 13, 2013	DoDots Licensing Solutions LLC
4			
5			

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading	
PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT

CLERK	(BY) DEPUTY CLERK	DATE

Copy 1—Upon initiation of action, mail this copy to Director **Copy 3—Upon termination of action, mail this copy to Director**
Copy 2—Upon filing document adding patent(s), mail this copy to Director **Copy 4—Case file copy**

Doc Code: PA..
Document Description: Power of Attorney

PTO/AIA/82A (07-13)
Approved for use through 03/31/2021. OMB 0651-0035
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

TRANSMITTAL FOR POWER OF ATTORNEY TO ONE OR MORE REGISTERED PRACTITIONERS

NOTE: This form is to be submitted with the Power of Attorney by Applicant form (PTO/AIA/82B) to identify the application to which the Power of Attorney is directed, in accordance with 37 CFR 1.5, unless the application number and filing date are identified in the Power of Attorney by Applicant form. If neither form PTO/AIA/82A nor form PTO/AIA82B identifies the application to which the Power of Attorney is directed, the Power of Attorney will not be recognized in the application.

Application Number	11932553
Filing Date	10/31/2007
First Named Inventor	Kembel
Title	DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES
Art Unit	2478
Examiner Name	LIN
Attorney Docket Number	0416-001B

SIGNATURE of Applicant or Patent Practitioner

Signature	/Peter G Korytnyk/	Date (Optional)	May 12, 2022
Name	Peter G Korytnyk	Registration Number	43400
Title (if Applicant is a juristic entity)			
Applicant Name (if Applicant is a juristic entity)			

NOTE: This form must be signed in accordance with 37 CFR 1.33. See 37 CFR 1.4(d) for signature requirements and certifications. If more than one applicant, use multiple forms.

☐ *Total of _____ forms are submitted.

This collection of information is required by 37 CFR 1.131, 1.32, and 1.33. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and 1.14. This collection is estimated to take 3 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

GENERAL POWER OF ATTORNEY BY APPLICANT AND/OR ASSIGNEE FOR UNITED STATES PATENT

U.S. DEPARTMENT OF COMMERCE
Patent and Trademark Office

I hereby revoke all previous powers of attorney given in the application identified in the attached transmittal letter (form PTO/AIA/82A or equivalent).

I hereby appoint Practitioner(s) associated with the following Customer Number for **PK Patent Law, LLC:**

41200

as my attorney(s) or agent(s), and to transact all business in the United States Patent and Trademark Office in connection with for the application referenced in the attached transmittal letter (form PTO/AIA/82A or equivalent). Please recognize or change the **correspondence address** for the application identified in the attached transmittal letter to the address associated with the above-mentioned **Customer Number**.

I am the:

Inventor or Joint Inventor.

Legal Representative of a Deceased or Legally Incapacitated Inventor

Assignee or Person to Whom the Inventor is Under an Obligation to Assign and am authorized to act on behalf of all assignees.

Person Who Otherwise Shows Sufficient Proprietary Interest (e.g., a petition under 37 CFR 1.46(b)(2) was granted in the application or is concurrently being filed with this document)

Assignee Name & Address:
DoDots Licensing Solutions, LLC
1701 Directors Blvd., Suite 300
Austin, TX 78744-1044

Signature of Applicant and/or Assignee:

Signature: _____ Date: May 6, 2022

Name (printed): David White

Telephone: (949) 939-0775

Title and Company: Sole Member, DoDots Licensing Solutions, LLC

Electronic Acknowledgement Receipt	
EFS ID:	45702716
Application Number:	11932553
International Application Number:	
Confirmation Number:	2604
Title of Invention:	DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES
First Named Inventor/Applicant Name:	John Albert Kembel
Customer Number:	909
Filer:	Peter George Korytnyk
Filer Authorized By:	
Attorney Docket Number:	043311-0381668
Receipt Date:	12-MAY-2022
Filing Date:	31-OCT-2007
Time Stamp:	20:37:13
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Power of Attorney	11932553POA.pdf	1966668 8d425d9f076e8d843512 06c38e4f3ce2bfc4375e	no	2
Warnings:					

Information:	
Total Files Size (in bytes):	1966668

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Trials@uspto.gov
571-272-7822

Paper 37
Date: January 5, 2021

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

LENOVO HOLDING COMPANY, INC., LENOVO (UNITED STATES) INC., AND MOTOROLA MOBILITY LLC,
Petitioner,

v.

DODOTS LICENSING SOLUTIONS LLC,
Patent Owner.

IPR2019-01279
Patent 8,510,407 B1

Before JAMES A. WORTH, AMBER L. HAGY, and SHARON FENICK, *Administrative Patent Judges.*

FENICK, *Administrative Patent Judge.*

JUDGMENT
Final Written Decision
Determining No Challenged Claims Unpatentable
35 U.S.C. § 318(a)
Granting In Part Petitioner's Motion to Strike
37 C.F.R. 42.5

I. INTRODUCTION

This is a Final Written Decision in this *inter partes* review challenging the patentability of claims 1, 8–13, and 20–24 of U.S. Patent No. 8,510,407 B1 (Ex. 1001, "the '407 patent"). We have jurisdiction under 35 U.S.C. § 6(b)(4). This Final Written Decision is issued pursuant to 35 U.S.C. § 318(a) and 37 C.F.R. § 42.73.

Petitioner bears the burden of proving unpatentability of the challenged claims, and the burden of persuasion never shifts to Patent Owner. *Dynamic Drinkware, LLC v. Nat'l Graphics, Inc.*, 800 F.3d 1375, 1378 (Fed. Cir. 2015). To prevail, Petitioner must prove unpatentability by a preponderance of the evidence. *See* 35 U.S.C. § 316(e) (2016); 37 C.F.R. § 42.1(d). Having reviewed the parties' arguments and supporting evidence, for the reasons discussed below, we find that Petitioner has not demonstrated by a preponderance of the evidence that any of claims 1, 8–13, and 20–24 are unpatentable. Additionally, for the reasons discussed below, in Section III.I, we grant-in-part Petitioner's Motion to Strike (Paper 31).

II. BACKGROUND AND SUMMARY

A. Procedural History

Lenovo Holding Company, Inc., Lenovo (United States) Inc., and Motorola Mobility LLC ("Petitioner") filed a Petition for an *inter partes* review of claims 1, 8–13, and 20–24 ("the challenged claims") of the '407 patent. Paper 2 ("Pet."). DoDots Licensing Solutions LLC ("Patent Owner") did not file a Preliminary Response. In view of the preliminary record, we concluded that Petitioner satisfied the burden, under 35 U.S.C. § 314(a), to show that there was a reasonable likelihood that Petitioner would prevail with respect to at least one of the challenged claims.

Accordingly, we instituted an *inter partes* review of all the challenged claims, on all of the asserted grounds. Paper 7 ("Inst. Dec.").

Patent Owner filed its Response. Paper 18 ("PO Resp."). Petitioner filed a Reply. Paper 21 ("Reply"). After receiving authorization, Patent Owner filed a motion to strike certain portions of the Reply and exhibits referenced and Petitioner filed an opposition. Paper 23; Paper 24. In an order relating to this motion to strike, we granted Patent Owner's motion and determined that a certain portion of the Reply was untimely and, along with certain exhibits, would not be considered. Paper 25, 4–7.

Patent Owner filed a Sur-reply. Paper 27 ("Sur-reply"). After receiving authorization, Petitioner filed a motion to strike Patent Owner's Sur-reply and a supplemental declaration filed with the Sur-reply. Paper 31 ("Pet. Mot." or "Petitioner's Motion to Strike"). Patent Owner filed an opposition. Paper 32 ("PO Opp."). We address Petitioner's Motion to Strike, *infra* Section III.I, and grant the motion with respect to the supplemental declaration, but not with respect to the Sur-reply.

On October 28, 2020, we held an oral hearing, the transcript of which is of record. Paper 36 ("Tr.").

B. Real Parties in Interest

Petitioner indicates that Lenovo Holding Company, Inc., Lenovo (United States) Inc., and Motorola Mobility LLC are the real parties-in-interest. Pet. 67.

Patent Owner indicates that DoDots Licensing Solutions, LLC is the real party-in-interest. Paper 4 (Patent Owner's Mandatory Notices), 2.

C. Related Matters

According to Petitioner and Patent Owner, the '407 patent at issue here is also asserted in *DoDots Licensing Solutions LLC v. Lenovo Holding*

Company, Inc. et al., No. 18-098-MN (D. Del.) ("the related district court action"). Pet. 67–68; Paper 4 (Patent Owner's Mandatory Notices), 2. Petitioner notes that the related district court action also involves U.S. Patent Nos. 9,369,545 and 8,020,083 ("the '083 patent"), and that Petitioner filed a petition for *inter partes* review of the patentability of certain claims in U.S. Patent No. 9,369,545 in IPR2019-00988 (final written decision finding no claims unpatentable issued, *see* IPR2019-00988, Paper 37 (Sept. 9, 2020)) and a petition for an *inter partes* review of the patentability of certain claims in U.S. Patent No. 8,020,083 (*inter partes* review instituted, *see* IPR2019-01278, Paper 8 (Jan. 24, 2020); final written decision pending). Pet. 68.

D. The '407 Patent

The title of the '407 patent is "Displaying Time-Varying Internet Based Data Using Application Media Packages." Ex. 1001, code (54). The '407 patent discloses, in part, a software component for accessing and displaying network content. *Id.* at code (57). A Networked Information Monitor (NIM) is a "fully configurable frame with one or more controls" with content optionally presented through the frame. *Id.* at 2:61–63, 5:21–24. When a NIM is opened by a user, the frame is presented in the user's display and network content is retrieved and presented in a viewer enclosed by the frame. *Id.* at 19:63–20:30. The network content may be identified via URLs included in the NIM definition. *Id.* at code (57), 20:24–27. The network content is time-varying, e.g., as in an image that varies over time. *Id.* at code (57). The Specification describes that the frame according to the invention "stands in contrast to present web browsers, which are branded by the browser vendor and which have limited means by which to alter the controls associated with the browser." *Id.* at 5:24–28.

E. Illustrative Claim

Claims 1 and 13 are the only independent claims among the challenged claims. Claim 1 is reproduced below with Petitioner's limitation designations added in brackets for ease of reference:

> 1. [1.Preamble] A client computing device configured to access content over a network, the client computing device comprising:
>
> [1.A] electronic storage configured to store networked information monitor template associated with a networked information monitor, [1.B] the networked information monitor template having therein a definition of a viewer graphical user interface having a frame within which time-varying content in a web browser-readable language may be presented on a display associated with the client computing device, wherein the frame of the viewer graphical user interface lacks controls for enabling a user to specify a network location at which content for the networked information monitor is available; and
>
> [1.C] one or more processors configured to execute one or more computer program modules, the one or more computer program modules being configured to access the networked information monitor defined by the networked information monitor template, wherein accessing the networked information monitor defined by the networked information monitor template results in:
>
> > [1.D] transmission, over a network to a web server at a network location, of a content request for content to be displayed within the frame of the viewer graphical user interface defined by the networked information monitor template;
> >
> > [1.E] reception, over the network from the web server at the network location, of content transmitted from the web server in response to the content request, the content being time-varying;
> >
> > [1.F] presentation, on the display, of the viewer graphical user interface defined by the networked

information monitor template outside of and separate from any graphical user interface of any other application; and

[1.G] presentation, on the display within the frame of the viewer graphical user interface defined by the networked information monitor, of the time-varying content received from the web server.

F. Evidence

Petitioner relies on the following references:

Reference	Description	Date	Exhibit
Van Hoff et al. ("Hoff")[1]	US 5,919,247	Issued July 6, 1999	Ex. 1004
Berg	Cliff Berg, *How Do I Create a Signed Castanet Channel?*, DR. DOBB'S JOURNAL, January 1, 1998	Jan. 1, 1998	Ex. 1009
Nazem	US 5,983,227	Issued Nov. 9, 1999	Ex. 1007
Fortin et al. ("Fortin")	US Patent Application Publication 2002/0023110 A1	Published Feb. 21, 2002	Ex. 1008
Razavi et al. ("Razavi")	US 6,401,134 B1	Issued June 4, 2002	Ex. 1006
Andersen	US 5,999,941	Issued Dec. 7, 1999	Ex. 1012

[1] The last name of the first-named inventor is Van Hoff, but, for consistency with the Petition, we refer to this patent as "Hoff."

G. Asserted Grounds

Petitioner asserts that claims 1, 8–13, and 20–24 would have been unpatentable on the following grounds:

Claim(s) Challenged	35 U.S.C. §[2]	Reference(s)/Basis
1, 9–13, 21–24	103	Hoff, Berg, Nazem or Admitted Prior Art ("APA")[3]
8, 20	103	Hoff, Berg, Nazem or APA, Fortin
1, 9–13, 21–24	103	Razavi, Andersen
8, 20	103	Razavi, Andersen, Fortin

III. ANALYSIS

A. Legal Standards

It is a petitioner's burden to demonstrate unpatentability. *See Dynamic Drinkware*, 800 F.3d at 1378 (citing *Tech. Licensing Corp. v. Videotek, Inc.*, 545 F.3d 1316, 1326–27 (Fed. Cir. 2008)).

A patent claim is unpatentable under 35 U.S.C. § 103(a) if the differences between the claimed subject matter and the prior art are "such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains." *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007). The question of obviousness is resolved based on underlying factual determinations, including: (1) the scope and content of the prior art; (2) any differences between the claimed subject matter and the prior art; (3) the level of skill in the art; and (4) objective evidence of nonobviousness, i.e.,

---

[2] Because the claims at issue have an effective filing date prior to the effective date of the AIA's amendments to 35 U.S.C. § 103, we apply the pre-AIA version of § 103 in this Decision.
[3] The Petition cites column 1, lines 56–67 of the '407 patent as the APA. Pet. 23–24, 26.

secondary considerations. *See Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966). "To satisfy its burden of proving obviousness, a petitioner cannot employ mere conclusory statements. The petitioner must instead articulate specific reasoning, based on evidence of record, to support the legal conclusion of obviousness." *In re Magnum Oil Tools Int'l, Ltd.*, 829 F.3d 1364, 1380 (Fed. Cir. 2016). We analyze the asserted grounds with the principles stated above in mind.

B. Level of Ordinary Skill in the Art

Petitioner asserts that a person of ordinary skill in the art "would have a bachelor's degree in Electrical or Computer Engineering, Computer Science, or a related field and have three or more years of experience" in the field. Pet. 8 (citing Ex. 1003 ("Madisetti Decl.") ¶ 44). Patent Owner does not provide an alternative description of the proper level of ordinary skill or comment on Petitioner's proposal.

Petitioner's proposal appears to be consistent with the prior art of record. *Okajima v. Bourdeau*, 261 F.3d 1350, 1355 (Fed. Cir. 2001) (explaining that specific findings regarding ordinary skill level are not required where the prior art itself reflects an appropriate level and a need for testimony is not shown). We adopt this proposal for this Final Written Decision.

C. Claim Construction

In an *inter partes* review based on a petition filed after November 13, 2018, the claims are construed

> using the same claim construction standard that would be used to construe the claim in a civil action under 35 U.S.C. [§] 282(b), including construing the claim in accordance with the ordinary and customary meaning of such claim as understood by one of

> ordinary skill in the art and the prosecution history pertaining to the patent.

See Changes to the Claim Construction Standard for Interpreting Claims in Trial Proceedings Before the Patent Trial and Appeal Board, 83 Fed. Reg. 51,340, 51,340, 51,358 (Oct. 11, 2018) (amending 37 C.F.R. § 42.100(b) effective November 13, 2018) (now codified at 37 C.F.R. § 42.100(b) (2019)); *see Phillips v. AWH Corp.*, 415 F.3d 1303, 1312–13 (Fed. Cir. 2005) (setting forth claim construction standard in civil actions). Any special definitions for claim terms must be set forth in the specification with reasonable clarity, deliberateness, and precision. *See In re Paulsen*, 30 F.3d 1475, 1480 (Fed. Cir. 1994).

1. "outside of and separate from any graphical user interface of any other application"

In the Petition, Petitioner requests that we construe one term, found in element 1.F of claim 1: that presentation of the viewer graphical user interface ("GUI") defined by the networked information monitor template be "outside of and separate from any graphical user interface of any other application." Pet. 4–8. Petitioner argues that the correct construction should be "so that the viewer [graphical user interface] is distinct from and not presented within the frame of any other GUI generated by another application." *Id.* Because our ultimate decision does not rest on the construction of this claim term, we do not construe it. *See Vivid Techs., Inc. v. Am. Sci. & Eng'g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999) ("[O]nly those terms need be construed that are in controversy, and only to the extent necessary to resolve the controversy."); *Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017) (applying *Vivid Techs.* in the context of an *inter partes* review).

2. "network information monitor"

To resolve Petitioner's challenge, the parties assert that two claim terms need to be construed: "network information monitor" and "network information monitor template" (or "NIM template"). PO Resp. 7–9; Reply 2–9; Sur-reply 2–7 ("The heart of the dispute between the parties with respect to claim construction relates to the term 'template' as used in the phrase 'networked information monitor . . . template.'"). Each term is introduced in each of the two independent claims from among the challenged claims. Ex. 1001, 42:28–64, 43:61–44:31. We address "network information monitor" first.

Patent Owner argues that "[t]he inventors defined NIMs in the specification as follows: 'As used herein, the term networked information monitor or NIM refers to a fully configurable frame with one or more controls; the frame through which content is optionally presented.'" PO Resp. 8 (quoting Ex. 1001, 5:21–24); Ex. 2004 ¶ 22. Petitioner agrees that "network information monitor" is defined in the Specification. Reply 2 (citing Ex. 1001, 2:59–61, 5:18–29); Tr. 14:20–23 (Petitioner's counsel stating that "[t]he patentee set out a clear and explicit definition of the term, network information monitor or NIM, said it refers to a fully configurable frame . . . with one or more controls."); *see* Ex. 1003 ¶ 33.

The definition of NIM provided in the Specification describes a frame "through which content is optionally presented." Ex. 1001, 5:21–24. Claim 1 also describes the subject of the definition in a NIM template alternatively as (a) a NIM (in 1.C), (b) "a viewer graphical user interface having a frame within which time-varying content in a web browser-readable language may be presented on a display associated with the client computing device" (in 1.B), or (c) a "viewer graphical user interface" (in 1.D, 1.F, 1.G). *Id.* at

42:28–64. Thus the definition of NIM provided in the Specification is consistent with the use in the claim. With respect to the word "optionally" in the Specification definition, we note that, in the claims, time-varying content is recited as being presented on the display within a frame (element 1.G and the last element of claim 13). *Id.*; *see also id.* at 43:61–44:31.

For these reasons, we determine that the correct construction of a NIM is "a fully configurable frame, with one or more controls, through which content is presented to the user," and this is the interpretation we apply here.

3. "network information monitor template"

a) The Parties' Contentions

Patent Owner asserts: "the NIM template is a data structure that defines the NIM; it is not the NIM and it is not an executable program (i.e., compiled code)." PO Resp. 8–9. Petitioner, while not presenting a proposed construction, argues that the Patent Owner's construction should be rejected as not supported by a definition in the '407 patent, and specifically argues that Patent Owner's proposed definition is flawed because it excludes an executable program from being a NIM template. Reply 1–9.

Patent Owner supports its proposed constructions with references to the Specification. Patent Owner quotes the Specification's description that "[e]ach NIM template defines the characteristics of a specific NIM." PO Resp. 8 (quoting Ex. 1001, 21:48–50) (emphasis omitted). With respect to "the definition of the NIM," Patent Owner quotes the Specification's statement that "the definition of the NIM is content, rather than compiled code" and cites additionally Figure 13, which "illustrates a data structure for a NIM definition." *Id.* (quoting Ex. 1001, 21:48–50, 3:63–65) (emphasis omitted). Patent Owner additionally quotes the Specification's description of the NIM definition as "fairly small in size" and "easily distributable"

because it "contains just enough information to define and initialize the NIM's components." *Id.* at 9 (quoting Ex. 1001, 33:55–65).

Patent Owner additionally relies on the testimony of its expert, Dr. Sacerdoti, regarding the plain and ordinary meaning of the word "template," and on Dr. Sacerdoti's testimony that, according to the '407 patent, a NIM template is not compiled code, and cannot be an executable application or applet. *Id.* at 8–9 (citing Ex. 2004 ¶¶ 29–36, 47–53).

Petitioner argues that the plain and ordinary meaning of the term should control, and that "nothing in the claims or specification limits a NIM template to a data structure excluding an executable program." Reply 1–3. Petitioner argues that Figure 13 of the '407 patent and the accompanying disclosure, cited by Patent Owner, are merely directed to a non-limiting example of a NIM template. *Id.* at 3–4. Petitioner also argues that Patent Owner's reliance on the language describing a NIM definition as "fairly small in size" is also exemplary. *Id.* at 4 (quoting Ex. 1001, 33:60). Petitioner notes that the "fairly small in size" NIM definition is further described in this portion of the '407 patent as "therefore easily distributable as an XML file or Blob (binary large object)." *Id.* (quoting Ex. 1001, 33:60–61). Petitioner contends that this binary large object "might contain any binary content, including executable code." *Id.* Thus, Petitioner argues, rather than finding indications that the patentee acted as lexicographer, Patent Owner improperly reads limitations from these examples, which the Petitioner contends are non-limiting, into its definition of "NIM template." *Id.* at 2–5.

Additionally, Petitioner argues that the Patent Owner's proposed construction improperly excludes disclosed embodiments. *Id.* at 5. Petitioner quotes the '407 patent's reference to "[a] second executable

module" that "defines a NIM frame for the NIM using the definition." *Id.* (quoting Ex. 1001, 3:8–10) (emphasis omitted). Petitioner also discusses the portion of the Specification that describes users sharing NIMS via database that stores "Share Module[s]" including "multiple . . . individual [NIM] definition modules" or "multiple packs of NIMs that have been shared . . . containing multiple NIM definition modules." Ex. 1001, 34:4–15, Fig. 24 (discussed and quoted-in-part at Reply 5–6). Petitioner argues that Dr. Sacerdoti "confirmed that a person of ordinary skill in the art would understand that '[a] module would be a chunk of code' and that an 'executable module' is 'executable computer code.'" Reply 6 (quoting Ex. 1019, 69:22–70:13).

Petitioner additionally argues that, in the related '083 patent, the claims describe one or more processors that "execute the first [NIM] template such that the graphical user interface of the first [NIM] is presented to a user." *Id.* at 7 (emphasis omitted); *see* Tr. 63:19–64:5, 71:6–14 (Patent Owner's and Petitioner's counsel each agreeing that "NIM template" in the '407 patent should be construed the same way in this *inter partes* review as it is in the *inter partes* review (IPR2019-01278) of claims of the '083 patent). Additionally Petitioner argues that Patent Owner's definition is contradicted by Patent Owner's statements in the prosecution history of other related patents and by Patent Owner's infringement allegations against Petitioner in the related district court action. *Id.* at 7–8 (citing Ex. 1016, 6–7, 14; Ex. 1015, 6, 7, 10–12). As discussed in our analysis below, Petitioner's arguments are not persuasive.

b) Construction

After considering the usage of "NIM template" in the claims, the Specification, and the prosecution history, and reviewing the parties'

contentions and evidence, we determine that a NIM template is a data structure that defines the characteristics of a NIM, including the NIM frame, view and control characteristics, and that excludes executable applications/compiled code.

Turning to the usage of NIM template in the claims, from the language of claim 1, we understand that the NIM template, stored in electronic storage of a client computing device (1.A), includes the definition of a NIM (1.C) / "viewer graphical user interface having a frame . . ." (1.B, 1.D, 1.F). Ex. 1001, 42:28–64; *see also id.* at 43:61–44:31 (similar limitations in claim 13). From our examination of the claims, we understand that the NIM template defines the characteristics of the viewer and the content on the viewer.

As to the prosecution history of the '407 patent, the parties did not identify, and we did not find, any portion of the prosecution history that is helpful in determining the meaning of NIM template.

The Specification of the '407 patent does address the meaning of NIM template. The Specification states the "NIMs allow a developer to provide an application feel without developing custom client applications" (Ex. 1001, 26:38–40), and the "NIM template defines the characteristics of a specific NIM, including fully configurable frame characteristics, viewer and control characteristics, and NIM content references" (*id.* at 6:66–7:2). *See also id.* at 21:12–14 ("The NIM definitions, as discussed above, include[] the NIM frame definition and the definition of the controls for filling the viewer within the frame with content.").

Figure 13 of the '407 patent is reproduced below.

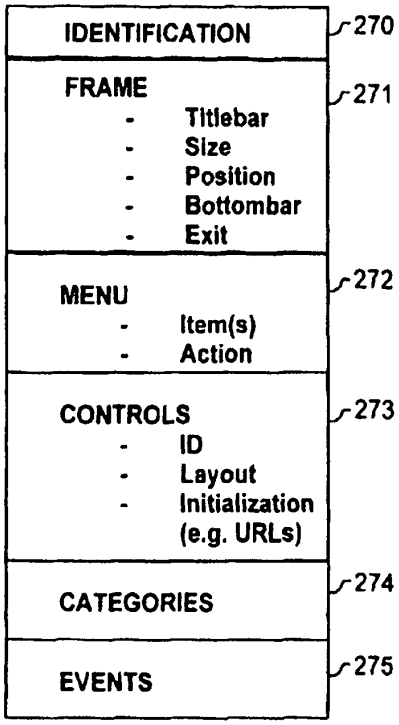


FIG. 13

Figure 13 "illustrates a data structure for a NIM definition." *Id.* at 21:32, 3:63–65. According to the Specification, in the exemplary embodiment of Figure 13 "the definition of a NIM includes tags that identify the NIM 270, define and configure the NIM frame 271, specify and layout the controls 273 in the NIM viewer, and specify parameters to initialize all the NIM's components with content or data." *Id.* at 21:61–65. This disclosure does not indicate that the data structure for the NIM template contains compiled code.

The Specification states that "a NIM is defined as a frame that contains a collection of controls, or functional units, such as a web rendering control or a GIF rendering control," and the NIM definitions "include all the

information needed to build a NIM frame and fill the NIM with NIM content." *Id.* at 21:33–35; 21:41–43. The Specification further states:

> In one embodiment, the NIM definitions are defined using Extensible Markup Language (XML), so that the NIM as a whole–the frame and the content within the viewer—is advantageously as flexible as standard web content. NIMs are extremely flexible, because ***the definition of the NIM is content, rather than compiled code***. The NIM definition defines the structure of the NIM, and everything that is visible in a NIM is based on standard Internet content, such as HTML, dHTML, or GIFs, and is referenced or pointed to by the NIM definition. An "application"–type NIM, such as a web calendar or web mail, may be changed by the user, by the content provider, or by other content, ***while advantageously avoiding the need to distribute and support a hard-coded compiled application***. The definition of a NIM thus includes everything that is needed for the NIM to be rendered and filled with Internet content.

Id. at 21:45–60 (emphases added). From this disclosure, we understand that the Specification prohibits the NIM template from using compiled code or an executable application. Our review of the Specification did not identify disclosure of use of compiled code or an executable application by the NIM template.

As discussed, Petitioner argues that the data structure illustrated in Figure 13 and the related discussion of Figure 13 merely describe one non-limiting embodiment, and that the Figure 13 embodiment merely describes implementing a NIM template as XML rather than compiled code. Reply 3–4 (citing *Innova/Pure Water, Inc. v. Safari Water Filtration Sys., Inc.*, 381 F.3d 1111, 1117 (Fed. Cir. 2004) ("[P]articular embodiments appearing in the written description will not be used to limit claim language that has broader effect.")). Petitioner's argument is not persuasive. Although we agree with Petitioner that the XML embodiment is one possible embodiment

of a NIM template, the Specification distinguishes the XML embodiment from embodiments using other languages for data structures, rather than distinguishing the XML embodiment from embodiments in which the template is compiled code. The Specification's broad statement that a NIM template is content, rather than compiled code or executable applications, applies to all NIM templates, not just one particular embodiment. The Specification does not describe any NIM template that uses compiled code or executable applications.

With respect to Petitioner's argument that a NIM template "might contain any binary content, including executable code," Petitioner's argument is not persuasive. Reply 4. The Specification refers to distribution of an NIM definition "as an XML file or Blob (binary large object)" and later, in the same context, to an "XML Blob." Ex. 1001, 33:61, 34:28. In addition, Petitioner's contention is attorney argument unsupported by persuasive expert testimony. *In re Geisler*, 116 F.3d 1465, 1470–71 (Fed. Cir. 1997) (argument by counsel cannot take the place of evidence); *In re Pearson*, 494 F.2d 1399, 1405 (CCPA 1974) ("Attorney's argument in a brief cannot take the place of evidence."). Unlike Patent Owner, Petitioner provided no declarant testimony probative to the issue of how the ordinary artisan would have understood and interpreted the relevant passages from the Specification and their bearing on the limitation in dispute.

With respect to the "second executable module" and "NIM definition modules" described in the Specification, Petitioner's arguments are again not persuasive. Reply 5–6. Petitioner does not identify, and we do not find, any portion of the Specification that discloses that the second executable module contains compiled code. In the context of the Specification, the "second executable module" described as acting to "define[] a NIM frame

for the NIM using the definition" is preceded by the description of a first executable module that "identif[ies] a definition of a Networked Information Monitor." Ex. 1001, 3:6–9. Although Petitioner argues that the second executable module *is* a NIM template, the second executable module is described in the Specification as "*using* the definition" that was identified by the first module. *Id.* (emphasis added). Petitioner only highlights the described action of the second executable module to "define a NIM frame" but does not explain the interaction with the first executable module, or support the assertion that the second executable module *is* a NIM template or that the discussed NIM definition modules are executable modules. *See* Reply 5–6. Although Petitioner cites Dr. Sacerdoti's deposition testimony relating to the word "module," Petitioner's contentions about the meaning of NIM template, and whether the "second executable module" or "NIM definition module" are NIM templates or executable modules, are attorney argument unsupported by persuasive expert testimony. *In re Geisler*, 116 F.3d at 1470–71. The Specification does not describe any NIM template that uses compiled code or executable applications. Rather, the Specification provides that "[e]ach NIM definition contains just enough information to define and initialize the NIM's components (NIM frame, controls, etc.)" and "the definition of the NIM is content, rather than compiled code." Ex. 1001, 33:55–57; Ex. 2004 ¶ 46.

Petitioner's argument regarding the claim term of the '083 patent describing "execut[ion of] the first networked information monitor template such that the graphical user interface of the first networked information monitor is presented to the user" amounts to one sentence, and does not address the context of the phrase in the claims of that patent, or present any testimony regarding whether one of ordinary skill in the art, reading those

claims, would determine that, in the cited claim 1 of the '083 patent, the networked information monitor template is described as executable code. Reply 7; Ex. 1013, 47:42–48:8. Petitioner's contentions about the meaning of the disputed terms in the context of the '083 claims and the prosecution history of a related application (Reply 6–7) are attorney argument unsupported by citations to persuasive intrinsic evidence or testimony from a witness competent to opine about how an ordinarily skilled artisan would understand the terms. *In re Geisler*, 116 F.3d at 1470–71. As such, Petitioner's argument is not persuasive.

Lastly, Petitioner's argument with respect to the positions taken by Patent Owner in the related district court action, in which, according to Petitioner, "Patent Owner has contended that each step of Claim 1 is performed by the use of" an "app" is again attorney argument. *See* Reply 8–9. Although Petitioner cites the Second Amended Complaint, that document, with respect to the '407 patent, describes generally that "Yogabook, in connection with the weather app" satisfies the limitations, and does not specifically describe what in that combination is mapped to the NIM template. Ex. 1015, 10–12; *see also id.* at 8–10 (discussing generally that "Lenovo's YogaBook also infringes Claim 1 of the '083 patent" without specific mapping to the NIM template). Petitioner's argument is not persuasive because it does not present any evidence about how these apps work and how they apply to the disputed NIM template. In addition, Petitioner's contentions about the meaning of the disputed terms and apps are attorney argument unsupported by persuasive evidence. *In re Geisler*, 116 F.3d at 1470–71. A party's infringement allegations do not override the meaning of the claims based on the intrinsic evidence.

For the reasons above, we determine that a NIM template is a data structure that defines the characteristics of a NIM, including the NIM frame, view and control characteristics, and that excludes executable applications/compiled code.

D. Obviousness over Hoff, Berg, and Nazem or APA

Petitioner contends that the subject matter of claims 1, 9–13, and 21–24 would have been obvious over the combined teachings of Hoff, Berg, and Nazem or APA. Pet. 20–37. We first describe the prior art that Petitioner seeks to combine to show that these claims are obvious, and then turn to the merits of the combination.

1. Hoff

Hoff is titled "Method for the Distribution of Code and Data Updates," and issued on July 6, 1999. Ex. 1004, codes (54), (45). Petitioner contends Hoff is prior art under 35 U.S.C. § 102(e). Pet. 8. Patent Owner does not dispute the prior-art status of Hoff.

Hoff describes a system and method for distributing software applications and data to thousands of clients over a network. Ex. 1004, code (57). In Hoff, an application is called a "channel," the server that distributes the channel is called the "transmitter," and the client receiving the channel is called the "tuner." *Id.* at code (57), 2:54–67. An end user, using a client/tuner, "subscribes" to a channel, the associated code and data are downloaded to the tuner, and then the channel can be executed many times without requiring further network access. *Id.* at code (57), 3:1–9. Any necessary software and data updates for the channel can be automatically downloaded and installed in the background at regular intervals by the tuner; the end user is not required to manually install software updates. *Id.* at code (57), 3:10–14, 5:4–28. Hoff asserts that this method of automatic

downloading of updates achieves for the client the same result as the broadcast distribution of software over a connection based network, but wherein the client initiates each update request without requiring any special broadcast networking infrastructure. *Id.* at code (57).

2. Berg[4]

Berg is an article by Clifford Berg titled "How Do I Create a Signed Castanet Channel?" Ex. 1009, 1. Petitioner asserts that Berg was published on January 1, 1998, and is prior art under 35 U.S.C. § 102(b). Pet. 9; *see* Ex. 1005 (Declaration of Clifford Berg).

Berg discloses using Marimba's Castanet, the "Java-centric (but not limited to Java) push technology," to distribute content to users. Ex. 1009, 1. Castanet is "a technology for deploying highly scalable applications, with centralized update and administration capabilities." *Id.* at 2. Berg states:

> Castanet consists of a client called the "tuner" and a server called the "transmitter." Channels are published by an administrator on a transmitter. A workstation that has a tuner installed can subscribe to any number of channels on any number of transmitters. The tuner updates the channel's content, which may consist of Java code, media, and other files, including binary code (signed channels can load native methods supported by DLLs). The tuner presents a user interface for managing

[4] The parties dispute whether Petitioner has met its burden of proof in the Petition that Berg was publicly accessible as of the critical date (April 26, 1999), and, therefore, available as a prior art printed publication. Pet. 9 (citing Ex. 1005 ¶¶ 2–3); *see also* Papers 22–25 (relating to and granting a motion to strike Petitioner's arguments in the Reply concerning this issue as not properly responsive to any argument or evidence in the PO Resp.); PO Resp. 12–14; Reply 9–11. We do not reach the dispute regarding the prior-art status of Berg because it is not necessary to our ultimate decision. Even if we accept Petitioner's view that Berg qualifies as prior art, we find that Petitioner has not established the unpatentability of any challenged patent claim.

> channels. The tuner interface is set apart from the interface constructed by any given channel (if the channel even has a user interface). This is unlike a browser, which imposes a user interface frame around any application that runs within it. Channels construct their own frames if and when they need them.

Id.

Berg describes, as an example, "a Castanet channel that opens a window on the screen, and allows a user to select from one of a small set of web locations to view." *Id.* at 3. The list of allowable locations is hard-coded into the program. *Id.* at 4. Once the user selects a location from the provided list, the application fetches data via the corresponding address and displays the content within a web-viewing window, implemented with an HTML renderer. *Id.* at 3, 4. The frame containing the choice selector (for choosing the web location) and the window in which the content is viewed also includes an "X" window-close control, which allows a user to close the frame. *Id.* at 4.

3. Nazem

Nazem is titled "Dynamic Page Generator" and issued on November 9, 1999. Ex. 1007, codes (54), (45). Petitioner contends Nazem is prior art under 35 U.S.C. § 102(e). Pet. 13.

Nazem describes pages containing live data arranged according to specific user preferences or recently used templates. Ex. 1007, code (57). Nazem's pages "[t]ypically" are "news pages." *Id.* Nazem describes the contents of such pages as "live data," which, in the case of news pages, might include "a custom selection of stock quotes, news headlines, sports scores, weather, and the like." *Id.*; *see also id.* at 1:61–2:14.

4. APA

Appellant refers to a portion of the Specification of the '407 patent as the APA. This portion, included in the Background of the Invention portion of the Specification, reads as follows:

> A user operating a client computer typically accesses the Internet by using a viewer application, such as a browser to view web content provided at a destination address, typically a web page. In this context, web content and web applications are designed to fill the entire web page. It is known to divide the web content into different regions of a single web page. For example, personalized web pages can be specified, such that a user views a variety of content sources in a single page, such as stock information, weather information, and sports information, which is aggregated at the server that delivers the web page to the user, who then views the aggregated content in a single web page.

Ex. 1001, 1:56–67 (cited at Pet. 23–24, 26).

5. Analysis of Claims 1, 9–13, and 21–24

In support of Petitioner's contention that claims 1, 9–13, and 21–24 are obvious over the combined teachings of Hoff, Berg, and Nazem or APA, Petitioner provides an element-by-element analysis showing where Petitioner contends each element of the challenged claims is in Hoff, Berg, or Nazem/APA. Pet. 20–37. Petitioner also provides supporting testimony by Dr. Madisetti (Ex. 1003). Patent Owner, relying on supporting testimony by Dr. Sacerdoti (Ex. 2004), argues that Petitioner does not show that the claims are unpatentable because, in part: (1) Hoff and Berg do not have a NIM template, (2) Hoff and Berg do not have a separate NIM and NIM template, and (3) objective evidence of non-obviousness such as commercial

success undermines Petitioner's arguments of unpatentability.[5] PO Resp. 1–3, 10–13, 15–16. For reasons set forth below, we determine that Petitioner has not shown that Hoff, Berg, and Nazem or APA teach or suggest the limitations of claims 1, 9–13, and 21–24.

a) Claims 1 and 13 – "network information monitor template"

Claim 1 recites a NIM and a NIM template in several limitations, including in limitations 1.A and 1.B:

> [1.A] electronic storage configured to store networked information monitor template associated with a networked information monitor, [1.B] the networked information monitor template having therein a definition of a viewer graphical user interface having a frame within which time-varying content in a web browser-readable language may be presented on a display associated with the client computing device, wherein the frame of the viewer graphical user interface lacks controls for enabling a user to specify a network location at which content for the networked information monitor is available[.]

Ex. 1001, 42:30–40. Claim 13 contains similar descriptions of a NIM and NIM template. *Id.* at 43:61–44:31.

Petitioner asserts that Hoff teaches the NIM template of claims 1 and 13 in the code and data downloaded to local storage when a channel is

---

[5] Patent Owner asserts that objective evidence of non-obviousness — commercial success — supports a finding of non-obviousness of the '407 patent. PO Resp. 15–16; Ex. 2003. We do not assess the objective evidence of non-obviousness because we find that Petitioner fails to carry its burden of demonstrating the unpatentability of any challenged claim based on the asserted prior art. *See, e.g., Hamilton Beach Brands, Inc. v. f'real Foods, LLC*, 908 F.3d 1328, 1343 (Fed. Cir. 2018) (holding, in affirming Board decision determining that petitioner had not shown unpatentability, that "objective indicia of nonobviousness" "need not [be] address[ed]" because the court "affirm[ed the] Board's findings regarding the failure of the prior art to teach or suggest all [claim] limitations").

subscribed to. Pet. 21–22 (citing Ex. 1004, 2:45–48, 2:56–67, 3:1–5, 4:47–49, Fig. 1B), 30–32. Petitioner asserts that Berg also discloses a tuner that is used to subscribe to channels, and that both Berg's and Hoff's channels include instructions and data used to create a GUI and display content therein, and that they therefore include a networked information monitor template. *Id.* at 22 (citing Ex. 1009, 2; Ex. 1003 ¶¶ 109–110). Petitioner asserts that limitation 1.B and a substantially identical limitation of claim 13 are taught by the combination of Berg and Nazem or APA, and that specifically, Berg discloses an application (Castanet channel) that includes a definition of an interface with a frame in which content is presented, and that the presented content may be rendered from HTML. *Id.* at 23 (citing Ex. 1003 ¶ 112; Ex. 1009, 3–4), 31.

Petitioner's arguments are not persuasive because Petitioner does not show persuasively that Hoff's and Berg's channel application comprises a data structure that defines the characteristics of a NIM, including the NIM frame, view and control characteristics, but that exclude executable applications/compiled code. *See supra* III.C.b (claim construction of "network information monitor template"). Petitioner admits that Hoff relates to distribution of software applications, such as Hoff's channels that are downloaded and installed and then executed as "a normal application." Pet. 8–9 (citing Ex. 1004, code (54), 1:54–67, 2:38–41, 4:8–10, 4:19–24, 4:51–57); Reply 8–9 (describing Hoff's channel code and data as the NIM template). Petitioner describes Berg's Castanet channel as an application that is deployed via Castanet and that can "do 'all the things a traditional installed application would do.'" Pet. 9–10 (citing Ex. 1009, 2–4, 6).

Petitioner's showing is also deficient because it fails to account for the fact that, as both Petitioner and Patent Owner agree, the NIM and the NIM

template are separate and distinct elements. Reply 2; PO Resp. 8; Tr. 16:14–19, 63:10–13. Neither Petitioner nor Dr. Madisetti identify what in Hoff and Berg might be a NIM template separate and distinct from the NIM. To the contrary, Petitioner argues that the channel applications (i.e., the NIM) contain the NIM template. *See, e.g.*, Pet. 23–24; Ex. 1003 ¶ 153 ("Specifically, the channel applications [disclosed in Hoff and Berg] contain instructions and data used to create a GUI and display content therein. Therefore, the 'channel' applications in Hoff and Berg include the claimed NIM template."). We also credit Dr. Sacerdoti's testimony that, under Petitioner's interpretation in which Hoff's and Berg's channel application (NIM) contains the claimed NIM template, Hoff and Berg still do not teach a NIM and a NIM template that are separate and distinct elements. Ex. 2004 ¶¶ 66 ("Even were a NIM template to be contained within the application that the petition identifies is the channel, the petition does not identify any data structure contained within the channel that could define the NIM. Not having identified any particular data structure, neither the petition nor Dr. Madisetti have identified a NIM definition that is disclosed by Hoff." (emphasis omitted)), 72 ("As with Hoff, neither Dr. Madisetti nor the petition asserts what in Berg might be the NIM template (channel template) that is distinct from the NIM (channel)." (emphasis omitted)).

Thus, for the reasons discussed above, we conclude that Petitioner has not demonstrated by a preponderance of the evidence that claims 1 and 13 would have been obvious over the combination of Hoff and Berg. Petitioner does not rely on Nazem or APA in a manner that remedies the foregoing deficiencies in Petitioner's showing as to the combination of Hoff and Berg.

b) Claims 9–12 and 21–24

Each of claims 9–12 and 21–24 depend, directly or indirectly, from independent claim 1 or independent claim 13, and, therefore, require a "NIM template." Ex. 1001, 43:5–44:31. Thus, for the same reasons discussed above with respect to Petitioner's challenge of claims 1 and 13, we also conclude that Petitioner has not demonstrated by a preponderance of the evidence that claims 9–12 and 21–24 would have been obvious over Hoff, Berg, and Nazem or APA.

c) Summary

For the reasons above, Petitioner has not shown that Hoff and Berg, either singly or in combination with each other, disclose a NIM template comprising a data structure separate and distinct from a NIM that defines the characteristics of a NIM, including the NIM frame, view and control characteristics, and that excludes executable applications/compiled code. As also noted above, Petitioner does not rely on Nazem or APA in a manner that remedies the foregoing deficiencies in Petitioner's showing as to the combination of Hoff and Berg. Thus, Petitioner has not shown by the preponderance of the evidence that claims 1, 9–13, and 21–24 are unpatentable over Hoff, Berg, and Nazem or APA.

E. Obviousness over Hoff, Berg, Nazem or APA, and Fortin

Petitioner also challenges the patentability of claims 8 and 20 over the combination of Hoff, Berg, Nazem or APA, and Fortin. Pet. 37–41. Claim 8 depends from independent claim 1, and claim 20 from independent claim 13, and therefore, each of claims 8 and 20 requires a "NIM template." Petitioner does not argue that the asserted combination remedies the deficiencies of Hoff and Berg discussed above. Thus, for the same reasons discussed above with respect to Petitioner's challenge of claims 1 and 13, we

also conclude that Petitioner has not demonstrated by a preponderance of the evidence that claims 8 and 13 would have been obvious over the asserted combination of Hoff, Berg, Nazem or APA, and Fortin.

F. Obviousness over Razavi and Andersen

Petitioner contends that the subject matter of claims 1, 9–13, and 21–24 would have been obvious over the combined teachings of Razavi and Andersen. Pet. 41–64. We first describe the prior art that Petitioner seeks to combine to show that these claims are obvious, and then turn to the merits of the combination.

1. Razavi

Razavi is titled "Detachable Java Applets," and issued on June 4, 2002. Ex. 1006, codes (54), (45). Petitioner contends Razavi is prior art under 35 U.S.C. § 102(e). Pet. 10.

Razavi describes "detaching Java applets from the constraints of the application such as a browser which provides the Java engine for executing those applets." Ex. 1006, code (57). In Razavi, a Java applet will be detachable from the application in which it was spawned and have the functionality of any ordinary application window running in the operating environment, including modifying the interface to suit the peculiarities of the applet. *Id.* at 4:41–56, 5:64–6:20. According to Razavi, the applet must be launched in the virtual machine of the application, and is detached if the user activates a detach user interface element. *Id.* at 5:48–63. Figure 4, reproduced below, is Razavi's illustration of a detached applet as rendered on a display screen. *Id.* at 7:8–9.

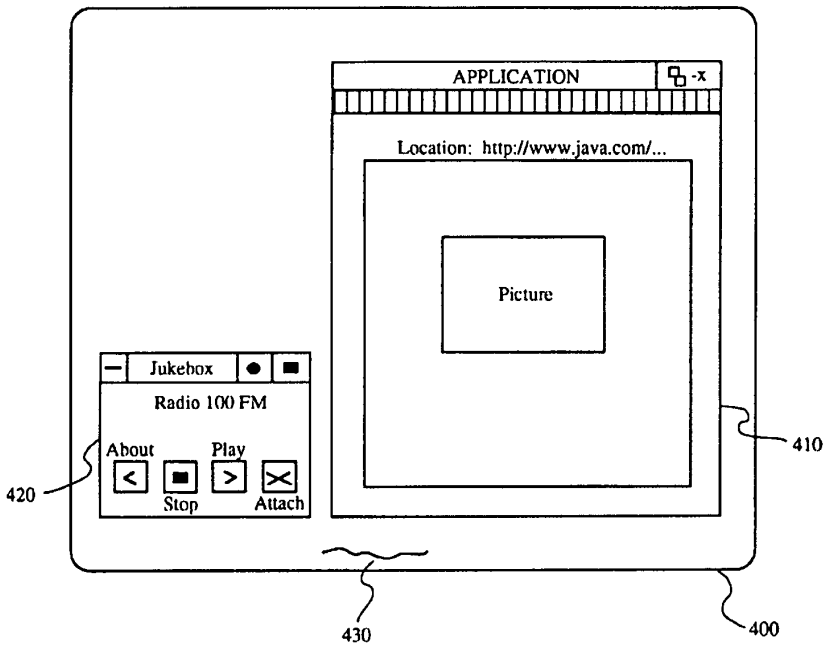


FIGURE 4

Figure 4 depicts application window 410 and detached applet 420 named "Jukebox" displayed on display 400. *Id.* at 7:10–12, 7:29–33. Application window 410 is running over an operating environment desktop 430, which also controls the general look-and-feel of the window for detached applet 420. *Id.* at 7:17–19, 17:39–41. Detached applet 420 contains applet controls "About," "Stop," and "Play," and a control called "Attach," which closes the detached applet 420 and "collapse[s] the applet back into application window 410." *Id.* at 7:34–35, 8:10–14.

The exemplary detached applet described in Razavi, the Jukebox applet, is a streaming audio applet that "creates VCR-type controls that allow the user to stop, play, or rewind the audio stream or to jump to a different selection" and optionally provides an image, such as a CD cover, to

be displayed, associated with each audio selection. *Id.* at 4:25–26, App'x A col. 1.

2. Andersen

Andersen is titled "Database Access Using Active Server Pages," and issued on December 7, 1999. Ex. 1012, codes (54), (45). Petitioner contends Andersen is prior art under 35 U.S.C. § 102(e). Pet. 12. Andersen describes an applet on a client computer that allows a database query to be embedded in an URL in order to obtain query results in HTML form. Ex. 1012, code (57), 3:5–29, 5:3–33. Andersen provides an example of a stock price quoting system implemented using a Java applet. *Id.* at 1:62–63, 1:67–2:30. The applet requests one or more stock prices from a server and the server responds with an HTML file into which data returned by the database is inserted in response to the query. *Id.* at 5:34–7:5. This allows an advantage over a system in which stock prices are available directly from an HTML page rather than from a database, because "storing the stock price data in static HTML pages can be quite inefficient, and data stored in this manner is not amenable to incremental updates and easy and efficient ad hoc searching and manipulation." *Id.* at 2:59–3:4.

3. Analysis of Claims 1, 9–13, and 21–24

In support of Petitioner's contention that claims 1, 9–13, and 21–24 are obvious over the combined teachings of Razavi and Andersen, Petitioner provides an element-by-element analysis showing where Petitioner contends each element of the challenged claims is in Razavi and Andersen. Pet. 41–64. Petitioner also provides supporting testimony by Dr. Madisetti (Ex. 1003). Patent Owner, relying on supporting testimony by Dr. Sacerdoti (Ex. 2004), argues that Petitioner does not show that claims 1, 9–13, and 21–24 are unpatentable because Razavi and Andersen teach accessing Java

applets through a web browser, and thus, as with the ground involving Hoff and Berg, Patent Owner therefore argues that: (1) the combination of Razavi and Andersen does not have a NIM template, and (2) the combination of Razavi and Andersen does not have a separate NIM and NIM template. PO Resp. 1–3, 10–15. Patent Owner additionally argues that one of ordinary skill would not have been motivated to combine Razavi and Andersen. *Id.* at 14. For reasons set forth below, we determine that Petitioner has not shown that Razavi and Andersen, either singly or in combination with each other, teaches or suggests the limitations of claims 1, 9–13, and 21–24.

a) Claims 1 and 13 – "network information monitor template"

Petitioner asserts that the client computing device of the preamble of claims 1 and 13 is taught by Razavi's and Andersen's disclosure of "similar computer-implemented methods in which an 'applet' is downloaded onto a client computing device, retrieves content from a server over a network, generates a GUI on a display, and presents the content within the GUI." Pet. 41–42 (citing Ex. 1006, 1:1–41; Ex. 1012, 1:10–2:30; Ex. 1003 ¶ 241), 58. Petitioner asserts that Razavi teaches the NIM template of claims 1 and 13 in Razavi's Java applets, downloaded in a pre-compiled state, and then executed at a client device. Pet. 43–44 (citing Ex. 1006, 1:9–25, 5:51–56, 7:29–33; Ex. 1003 ¶¶ 246–248), 58–60.

Petitioner's arguments are not persuasive because Petitioner does not show persuasively that Razavi's and Andersen's applets comprise a data structure that defines the characteristics of a NIM, including the NIM frame, view and control characteristics, but that excludes executable applications/compiled code. Razavi relates to distribution of executable platform-independent Java applets. Ex. 1006, 1:13–16 (platform-independent applets are downloaded as pre-compiled Java byte-codes),

1:26–30 (applet runs on a platform's virtual machine capable of executing applet code), 3:13–15; Ex. 1003 ¶ 98; Ex. 2004 ¶ 83.

Petitioner's showing is also deficient because it fails to account for the fact that, as discussed previously, both Petitioner and Patent Owner agree that the NIM and the NIM template are separate and distinct elements. Reply 2; PO Resp. 8; Tr. 18:1–7, 56:18–19, 69:10–11. However, neither Petitioner nor Dr. Madisetti identify what in Razavi or Andersen might be a NIM template separate and distinct from the NIM. To the contrary, Petitioner argues that the received applet in Razavi contains definitions for the GUI generated by the applet. Pet. 44–45; Ex. 1003 ¶ 249. In addition, Dr. Sacerdoti's unrebutted testimony is that a person of ordinary skill in the art would understand that Razavi's and Andersen's Java applets disclose no separate definition of or template for applets that might teach or suggest a distinct NIM template. Ex. 2001 ¶ 91; *see id.* ¶ 83 ("Applets and applications were known in the art to be executable code."). Thus, for the reasons discussed above, we conclude that Petitioner has not demonstrated by a preponderance of the evidence that claims 1 and 13 would have been obvious over the asserted prior art.

b) Claims 9–12 and 21–24

Each of claims 9–12 and 21–24 depends, directly or indirectly, from independent claim 1 or independent claim 13, and, therefore, requires a "NIM template." Ex. 1001, 43:5–44:31. Thus, for the same reasons discussed above with respect to Petitioner's challenge of claims 1 and 13, we also conclude that Petitioner has not demonstrated by a preponderance of the evidence that claims 9–12 and 21–24 would have been obvious over Razavi and Andersen.

c) Summary

For the reasons above, Petitioner has not shown that Razavi and Andersen, either singly or in combination with each other, disclose a NIM template comprising a data structure that defines the characteristics of a NIM, including the NIM frame, view and control characteristics, and that excludes executable applications/compiled code. Thus, Petitioner has not shown by the preponderance of the evidence that claims 1, 9–13, and 21–24 are unpatentable over Razavi and Andersen.

G. Obviousness over Razavi, Andersen, and Fortin

Petitioner also challenges the patentability of claims 8 and 20 over the combination of Razavi, Andersen, and Fortin. Pet. 64–67. Claim 8 depends from independent claim 1, and claim 20 from independent claim 13, and therefore, each of claims 8 and 20 requires a "NIM template." Petitioner does not argue that the asserted combination remedies the deficiencies of Razavi and Andersen discussed above. Thus, for the same reasons discussed above with respect to Petitioner's challenge of claims 1 and 13, we also conclude that Petitioner has not demonstrated by a preponderance of the evidence that claims 8 and 13 would have been obvious over the asserted combination of Razavi, Andersen, and Fortin.

H. Patent Owner's "'Secret' Prior Art" Argument

Patent Owner argues that, because Hoff and Razavi were published after the critical date for the '407 patent (April 26, 1999), they are "'secret' prior art" because they were not available to one of ordinary skill in the art as of the critical date. PO Resp. 16–17. Patent Owner acknowledges that, under current Federal Circuit precedent, these patents are considered prior art despite not being publicly accessible as of the critical date, but Patent Owner contends that the law should be changed. *Id.*

Because we must follow the law as it stands, we decline to address Patent Owner's arguments.

I. Petitioner's Motion to Strike

As discussed *supra*, Petitioner's Motion to Strike requested that we strike Patent Owner's Sur-reply and a supplemental declaration of Dr. Sacerdoti (Ex. 2007, "Supplemental Declaration") filed with the Sur-reply. Pet. Mot. 1. Petitioner argues that filing of the Supplemental Declaration contravenes our Practice Guide governing the conduct of this proceeding that a sur-reply "may not be accompanied by new evidence other than deposition transcripts of the cross-examination of any reply witness." *Id.* (quoting Patent Trial and Appeal Board Consolidated Trial Practice Guide November 2019 ("Consolidated TPG") 73). Patent Owner argues that the Consolidated TPG is only a guide and that its statements are not controlling or binding rules. PO Opp. 1. Patent Owner additionally argues, generally, that the Supplemental Declaration was necessary to address "arguments raised" in Petitioner's Reply, providing only one general example: that "Petitioners argued in their Reply that Patent Owner's Response was inconsistent with positions Patent Owner has taken in the district court litigation." *Id.* at 2–3. However, the Supplemental Declaration only discusses this issue in one paragraph, its last one.

As we indicated during the hearing (Tr. 4:18–5:2), we grant-in-part the Motion to Strike, with respect to striking the Supplemental Declaration, which constitutes new evidence filed with the Sur-reply. With respect to the Sur-reply itself, the Consolidated TPG notes that "[t]he Board is not required to attempt to sort proper from improper portions of the reply or sur-reply" and that a reply that raises a new issue might not be considered. Consolidated TPG 74. However, "striking the entirety or a portion of a

party's brief is an exceptional remedy that the Board expects will be granted rarely," and we are capable of disregarding any new issues exceeding the proper scope of reply. *Id.* at 80. We determine that this is not a case requiring that exceptional remedy, and that the Sur-reply need not be stricken.

Thus, we grant-in-part and deny-in-part the Motion to Strike.

IV. CONCLUSION

For the reasons expressed above, we determine that Petitioner has not proven by a preponderance of the evidence that the challenged claims are unpatentable.

In summary:

Claims	35 U.S.C. §	Reference(s)/Basis	Claims Shown Unpatentable	Claims Not shown Unpatentable
1, 9–13, 21–24	103	Hoff, Berg, Nazem or APA		1, 9–13, 21–24
8, 20	103	Hoff, Berg, Nazem or APA, Fortin		8, 20
1, 9–13, 21–24	103	Razavi, Andersen		1, 9–13, 21–24
8, 20	103	Razavi, Andersen, Fortin		8, 20
Overall Outcome				1, 8–13, 20–24

V. ORDER

In consideration of the foregoing, it is hereby:

ORDERED that claims 1, 8–13, and 20–24 of the '407 patent have not been shown to be unpatentable;

ORDERED that Petitioner's Motion to Strike (Paper 31) is granted-in-part as discussed above; and

FURTHER ORDERED that, because this is a Final Written Decision, parties to the proceeding seeking judicial review of the Decision must comply with the notice and service requirements of 37 C.F.R. § 90.2.

FOR PETITIONER:

John C. Alemanni
Matthew J. Meyer
KILPATRICK TOWNSEND & STOCKTON LLP
jalemanni@kilpatricktownsend.com
mmeyer@kilpatricktownsend.com

FOR PATENT OWNER:

Lewis E. Hudnell, III
HUDNELL LAW GROUP P.C.
lewis@hudnellaw.com

Trials@uspto.gov
571-272-7822

Paper 7
Date: January 9, 2020

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

LENOVO HOLDING COMPANY, INC., LENOVO (UNITED STATES) INC., AND MOTOROLA MOBILITY LLC,
Petitioner,

v.

DODOTS LICENSING SOLUTIONS LLC,
Patent Owner.

IPR2019-01279
Patent 8,510,407 B1

Before JAMES A. WORTH, AMBER L. HAGY, and SHARON FENICK, *Administrative Patent Judges.*

FENICK, *Administrative Patent Judge.*

DECISION
Granting Institution of *Inter Partes* Review
35 U.S.C. § 314, 37 C.F.R. § 42.4

I. INTRODUCTION

A. Background and Summary

Lenovo Holding Company, Inc., Lenovo (United States) Inc., and Motorola Mobility LLC ("Petitioner") filed a Petition for an *inter partes* review of claims 1, 8–13, and 20–24 of U.S. Patent No. 8,510,407 B1

(Ex. 1001, "the '407 patent"). Paper 2 ("Pet."). DoDots Licensing Solutions LLC ("Patent Owner") did not file a Preliminary Response.

Under 35 U.S.C. § 314 and 37 C.F.R. § 42.4(a), we have authority to determine whether to institute an *inter partes* review. If an *inter partes* review is instituted, a final written decision under 35 U.S.C. § 318(a) must decide the patentability of all claims challenged in the petition. *SAS Inst., Inc. v. Iancu*, 138 S. Ct. 1348, 1359–60 (2018).

Upon considering the Petition and the evidence of record, we determine that Petitioner has demonstrated a reasonable likelihood of prevailing in showing the unpatentability of at least one of the challenged claims. For the reasons described below, we institute an *inter partes* review of claims 1, 8–13, and 20–24 of the '407 patent with respect to all grounds in the Petition.

B. Real Parties in Interest

Petitioner indicates that Lenovo Holding Company, Inc., Lenovo (United States) Inc., and Motorola Mobility LLC are the real parties-in-interest. Pet. 67.

Patent Owner indicates that DoDots Licensing Solutions, LLC is the real party-in-interest. Paper 4 (Patent Owner's Mandatory Notices), 2.

C. Related Matters

According to Petitioner and Patent Owner, the '407 patent at issue here is also asserted in *DoDots Licensing Solutions LLC v. Lenovo Holding Company, Inc. et al.*, Case No. 18-098-MN (D. Del.). Pet. 67–68; Paper 4 (Patent Owner's Mandatory Notices), 2. Petitioner notes that that case also involves U.S. Patent Nos. 9,369,545 and 8,020,083, and that Petitioner filed a petition for *inter partes* review of the patentability of claims in U.S. Patent No. 9,369,545 in IPR2019-00988 (*inter partes* review instituted, *see*

IPR2019-00988, Paper 7 (Sept. 10, 2019)) and a petition for an *inter partes* review of the patentability of claims in U.S. Patent No. 8,020,083 (decision on institution pending). Pet. 68.

D. The '407 Patent

The title of the '407 patent is "Displaying Time-Varying Internet Based Data Using Application Media Packages." Ex. 1001, code (54). The '407 patent discloses, in part, a software component for accessing and displaying network content. *Id.* at code (57). A Networked Information Monitor (NIM) is a "fully configurable frame with one or more controls" with content optionally presented through the frame. *Id.* at 2:61–63, 5:21–24. When a NIM is opened by a user, the frame is presented in the user's display and network content is retrieved and presented in a viewer enclosed by the frame. *Id.* at 19:63–20:30. The network content may be identified via URLs included in the NIM definition. *Id.* at code (57), 20:24–27. The network content is time-varying, e.g. as in an image that varies over time. *Id.* at code (57). The Specification describes that the frame according to the invention "stands in contrast to present web browsers, which are branded by the browser vendor and which have limited means by which to alter the controls associated with the browser." *Id.* at 5:24–28.

E. Illustrative Claim

The challenged claims are claims 1, 8–13, and 20–24. Claims 1 and 13 are the only independent claims among the challenged claims. Claim 1 is reproduced below with Petitioner's bracketed limitation designations added for ease of reference:

1. [1.Preamble] A client computing device configured to access content over a network, the client computing device comprising:

[1.A] electronic storage configured to store networked information monitor template associated with a networked information monitor, [1.B] the networked information monitor template having therein a definition of a viewer graphical user interface having a frame within which time-varying content in a web browser-readable language may be presented on a display associated with the client computing device, wherein the frame of the viewer graphical user interface lacks controls for enabling a user to specify a network location at which content for the networked information monitor is available; and

[1.C] one or more processors configured to execute one or more computer program modules, the one or more computer program modules being configured to access the networked information monitor defined by the networked information monitor template, wherein accessing the networked information monitor defined by the networked information monitor template results in:

[1.D] transmission, over a network to a web server at a network location, of a content request for content to be displayed within the frame of the viewer graphical user interface defined by the networked information monitor template;

[1.E] reception, over the network from the web server at the network location, of content transmitted from the web server in response to the content request, the content being time-varying;

[1.F] presentation, on the display, of the viewer graphical user interface defined by the networked information monitor template outside of and separate from any graphical user interface of any other application; and

[1.G] presentation, on the display within the frame of the viewer graphical user interface defined by the networked information monitor, of the time-varying content received from the web server.

F. Evidence

Petitioner relies on the following references:

Reference	Description	Date	Exhibit
Van Hoff et al. ("Hoff")[1]	US 5,919,247	Issued July 6, 1999	Ex. 1004
Berg	Cliff Berg, *How Do I Create a Signed Castanet Channel?*, DR. DOBB'S JOURNAL, January 1, 1998	Jan. 1, 1998	Ex. 1009
Nazem	US 5,983,227	Issued Nov. 9, 1999	Ex. 1007
Fortin et al. ("Fortin")	US Patent Application Publication 2002/0023110 A1	Published Feb. 21, 2002	Ex. 1008
Razavi et al. ("Razavi")	US 6,401,134 B1	Issued June 4, 2002	Ex. 1006
Andersen	US 5,999,941	Issued Dec. 7, 1999	Ex. 1012

Petitioner also relies on the Declaration of Dr. Vijay K. Madisetti. Ex. 1003 ("Madisetti Decl.").

G. Prior Art and Asserted Grounds

Petitioner asserts that claims 1, 8–13, and 20–24 would have been unpatentable on the following grounds:

Claim(s) Challenged	35 U.S.C. §	Reference(s)/Basis
1, 9–13, 21–24	103	Hoff, Berg, Nazem or Admitted Prior Art ("APA")[2]
8, 20	103	Hoff, Berg, Nazem or APA, Fortin
1, 9–13, 21–24	103	Razavi, Andersen
8, 20	103	Razavi, Andersen, Fortin

[1] The last name of the first-named inventor is Van Hoff, but, for consistency with the Petition, we refer to this patent as "Hoff."
[2] The Petition cites column 1, lines 56–67 of the '407 patent as the APA. Pet. 23–24, 26.

II. ANALYSIS

A. Legal Standards

It is a petitioner's burden to demonstrate unpatentability. *See Dynamic Drinkware, LLC v. Nat'l Graphics, Inc.*, 800 F.3d 1375, 1378 (Fed. Cir. 2015) (citing *Tech. Licensing Corp. v. Videotek, Inc.*, 545 F.3d 1316, 1326–27 (Fed. Cir. 2008)).

A patent claim is unpatentable under 35 U.S.C. § 103(a) if the differences between the claimed subject matter and the prior art are "such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains." *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007). The question of obviousness is resolved based on underlying factual determinations, including: (1) the scope and content of the prior art; (2) any differences between the claimed subject matter and the prior art; (3) the level of skill in the art; and (4) objective evidence of nonobviousness, i.e., secondary considerations. *See Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966). "To satisfy its burden of proving obviousness, a petitioner cannot employ mere conclusory statements. The petitioner must instead articulate specific reasoning, based on evidence of record, to support the legal conclusion of obviousness." *In re Magnum Oil Tools Int'l, Ltd.*, 829 F.3d 1364, 1380 (Fed. Cir. 2016). We analyze the asserted grounds with the principles stated above in mind.

B. Level of Ordinary Skill in the Art

Petitioner asserts that a person of ordinary skill in the art "would have a bachelor's degree in Electrical or Computer Engineering, Computer Science, or a related field and have three or more years of experience" in the field. Pet. 8 (citing Madisetti Decl. ¶ 44).

Petitioner's proposal appears to be consistent with the prior art of record. *Okajima v. Bourdeau*, 261 F.3d 1350, 1355 (Fed. Cir. 2001) (explaining that specific findings regarding ordinary skill level are not required where the prior art itself reflects an appropriate level and a need for testimony is not shown). For the purposes of this Decision on Institution, we adopt Petitioner's yet-unchallenged asserted level of ordinary skill solely to determine whether there is a reasonable likelihood that the Petitioner would prevailing in showing the unpatentability of at least one of the challenged claims.

C. Claim Construction

In an *inter partes* review based on a petition filed after November 13, 2018,[3] the claims are construed

> using the same claim construction standard that would be used to construe the claim in a civil action under 35 U.S.C. [§] 282(b), including construing the claim in accordance with the ordinary and customary meaning of such claim as understood by one of ordinary skill in the art and the prosecution history pertaining to the patent.

37 C.F.R. § 42.100(b) (2019); *see Phillips v. AWH Corp.*, 415 F.3d 1303, 1312–13 (Fed. Cir. 2005) (setting forth claim construction standard in civil actions). Any special definitions for claim terms must be set forth in the specification with reasonable clarity, deliberateness, and precision. *See In re Paulsen*, 30 F.3d 1475, 1480 (Fed. Cir. 1994).

Petitioner requests that we construe one term from claim 1: that presentation of the viewer graphical user interface ("GUI") defined by the

[3] *See* Changes to the Claim Construction Standard for Interpreting Claims in Trial Proceedings Before the Patent Trial and Appeal Board, 83 Fed. Reg. 51,340 (Oct. 11, 2018) (amending 37 C.F.R. § 42.100(b) effective November 13, 2018).

networked information monitor template be "outside of and separate from any graphical user interface of any other application." Pet. 4–8. Petitioner argues that the correct construction should be "so that the viewer [graphical user interface] is distinct from and not presented within the frame of any other GUI generated by another application." *Id.*

We determine that no claim term requires express construction for purposes of this Decision. *See Vivid Techs., Inc. v. Am. Sci. & Eng'g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999) ("[O]nly those terms need be construed that are in controversy, and only to the extent necessary to resolve the controversy."); *Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017) (applying *Vivid Techs.* in the context of an *inter partes* review).

D. Obviousness over Hoff, Berg, and Nazem or APA

Petitioner contends that the subject matter of claims 1, 9–13, and 21–24 would have been obvious over the combined teachings of Hoff, Berg, and Nazem or APA. Pet. 20–37. The factual and legal contentions presented in the Petition are unrebutted at this stage of the proceeding. We have reviewed the information provided by Petitioner, including the relevant portions of the supporting Madisetti Declaration. Based on the current record, for reasons set forth below, we are persuaded that Petitioner has demonstrated a reasonable likelihood of prevailing on this obviousness challenge.

1. Hoff

Hoff is titled "Method for the Distribution of Code and Data Updates," and issued on July 6, 1999. Ex. 1004, codes (54), (45). Petitioner contends Hoff is prior art under 35 U.S.C. § 102(e). Pet. 8.

Hoff describes a system and method for distributing software applications and data to thousands of clients over a network. Ex. 1004, code (57). In Hoff, an application is called a "channel," the server that distributes the channel is called the "transmitter," and the client receiving the channel is called the "tuner." *Id.* at code (57), 2:54–67. An end-user, using a client/tuner, "subscribes" to a channel, the associated code and data is downloaded to the tuner, and then the channel can be executed many times without requiring further network access. *Id.* at code (57), 3:1–9. Any necessary software and data updates for the channel can be automatically downloaded and installed in the background at regular intervals by the tuner; the end-user is not required to manually install software updates. *Id.* at code (57), 3:10–14, 5:4–28. Hoff asserts that this method of automatic downloading of updates achieves for the client the same result as the broadcast distribution of software over a connection based network, but wherein the client initiates each update request without requiring any special broadcast networking infrastructure. *Id.* at code (57).

2. Berg

Berg is an article by Clifford Berg titled "How Do I Create a Signed Castanet Channel?". Ex. 1009, 1. Petitioner asserts that Berg was published on January 1, 1998, and is prior art under 35 U.S.C. § 102(b). Pet. 9; *see* Ex. 1005 (Declaration of Clifford Berg). We conclude, based on the indicia on the face of Berg, and in light of the additional evidence cited by Petitioner, that there is a reasonable likelihood that Berg qualifies as a printed publication. *See Hulu, LLC v. Sound View Innovs.*, LLC, IPR2018-01039, Paper 29 at 13, 17–18 (PTAB Dec. 20, 2019) (precedential).

Berg discloses using Marimba's Castanet, the "Java-centric (but not limited to Java) push technology," to distribute content to users. Berg, 1.

Castanet is "a technology for deploying highly scalable applications, with centralized update and administration capabilities." *Id.* at 2. Berg states that:

> Castanet consists of a client called the "tuner" and a server called the "transmitter." Channels are published by an administrator on a transmitter. A workstation that has a tuner installed can subscribe to any number of channels on any number of transmitters. The tuner updates the channel's content, which may consist of Java code, media, and other files, including binary code (signed channels can load native methods supported by DLLs). The tuner presents a user interface for managing channels. The tuner interface is set apart from the interface constructed by any given channel (if the channel even has a user interface). This is unlike a browser, which imposes a user interface frame around any application that runs within it. Channels construct their own frames if and when they need them.

Id.

Berg describes, as an example, "a Castanet channel that opens a window on the screen, and allows a user to select from one of a small set of web locations to view." *Id.* at 3. The list of allowable locations is hard-coded into the program. *Id.* at 4. Once the user selects a location from the provided list, the application fetches data via the corresponding address and displays the content within a web-viewing window, implemented with an HTML renderer. *Id.* at 3, 4. The frame containing the choice selector (for choosing the web location) and the window in which the content is viewed also includes an "X" window-close control, which allows a user to close the frame. *Id.* at 4.

3. Nazem

Nazem is titled "Dynamic Page Generator" and issued on November 9, 1999. Ex. 1007, codes (54), (45). Petitioner contends Nazem is prior art under 35 U.S.C. § 102(e). Pet. 13.

Nazem describes pages containing live data arranged according to specific user preferences or recently used templates. Ex. 1007, code (57). Nazem's pages "[t]ypically" are "news pages." *Id.* Nazem describes the contents of such pages as "live data," which, in the case of news pages, might include "a custom selection of stock quotes, news headlines, sports scores, weather, and the like." *Id.*; *id.* at 1:61–2:14.

4. APA

Appellant refers to a portion of the Specification of the '407 patent as the APA. This portion, included in the Background of the Invention portion of the Specification, reads as follows:

> A user operating a client computer typically accesses the Internet by using a viewer application, such as a browser to view web content provided at a destination address, typically a web page. In this context, web content and web applications are designed to fill the entire web page. It is known to divide the web content into different regions of a single web page. For example, personalized web pages can be specified, such that a user views a variety of content sources in a single page, such as stock information, weather information, and sports information, which is aggregated at the server that delivers the web page to the user, who then views the aggregated content in a single web page.

'407 patent, 1:56–67 (cited at Pet. 23–24, 26).

5. Analysis of Claim 1

a) Preamble

The preamble of claim 1 recites a "client computing device configured to access content over a network." Petitioner asserts that this preamble is taught by Hoff's and Berg's disclosure of "similar computer implemented methods in which a 'tuner' located on a client device is used to download a 'channel' application from 'transmitter' located on a server over a network." Pet. 20 (citing Ex. 1004, code (57); Ex. 1009, 1; Madisetti Decl. ¶ 104). Reproduced below is Figure 1A of Hoff, as annotated by Petitioner:

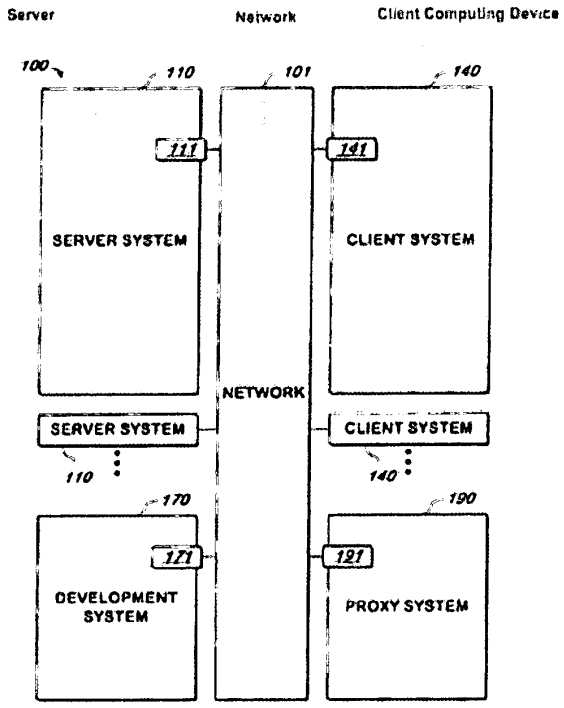


FIG. 1A

Id. at 21. Petitioner argues that the client system 140 is a client system used to obtain content "such as the channel disclosed in Berg." *Id.* at 20 (citing Ex. 1004, 3:31–49, 3:66–4:2; Madisetti Decl. ¶ 105).

b) Limitation 1.A

Claim limitation 1.A recites that the client computing device comprises "electronic storage configured to store networked information monitor template associated with a networked information monitor." Petitioner asserts that this limitation is taught by Hoff, in which a client device includes a storage system, and code and data are downloaded to local storage when a channel is subscribed to. Pet. 21–22 (citing Ex. 1004, 2:45–48, 2:56–67, 3:1–5, 4:47–49, Fig. 1B). Petitioner asserts that Berg also discloses a tuner that is used to subscribe to channels, and that both Berg's and Hoff's channels include instructions and data used to create a GUI and display content therein, satisfying the networked information monitor template as discussed with reference to later limitations. Pet. 22 (citing Madisetti Decl. ¶¶ 109–110).

c) Limitation 1.B

Claim limitation 1.B recites "the networked information monitor template having therein a definition of a viewer graphical user interface having a frame within which time-varying content in a web browser-readable language may be presented on a display associated with the client computing device, wherein the frame of the viewer graphical user interface lacks controls for enabling a user to specify a network location at which content for the networked information monitor is available." Petitioner asserts that this limitation is taught by the combination of Berg and Nazem or APA. Pet. 23–24. Petitioner argues that Berg discloses an application that includes a definition of an interface with a frame in which content is presented, and that the presented content may be rendered from HTML. *Id.* at 23 (citing Ex. 1009, 3–4). Petitioner further argues that Nazem or the APA discloses that HTML content might include time-varying content such

as stock quotes, weather data, and sports scores. *Id.* at 23–24 (citing Ex. 1007, 1:60–2:14; Ex. 1001, 1:56–67).

With respect to the frame of the GUI "lack[ing] controls for enabling a user to specify a network location at which content for the networked information monitor is available," Petitioner argues that while Berg's application indicates that several URLs are available for a user to select, one of ordinary skill would have been motivated to make a modification so only one URL would be available, and would have had a reasonable expectation of success in such a modification. Pet. 24, 35–36 (citing Madisetti Decl. ¶¶ 116–117, 218–219). Petitioner notes that Berg's channel is described as a solution for a company that wishes to provide users with "web access, but in a controlled way," and that to additionally control access, users of a channel might be restricted to only one URL. Ex. 1009, 3; Pet. 35–36 (citing Ex. 1009, 3, 6; Madisetti Decl. ¶¶ 218–219). We note that Petitioner does not indicate whether a user control must be provided for a "graphical user interface" according to claim 1, but that in modifying Berg's channel "by not including controls for manually navigating a network," Berg's described "window-close event" user control would still be included. *See* Ex. 1009, 4; Madisetti Decl. ¶¶ 93, 129.

d) Limitation 1.C

Claim limitation 1.C recites "one or more processors configured to execute one or more computer program modules, the one or more computer program modules being configured to access the networked information monitor defined by the networked information monitor template." Petitioner asserts that this limitation is taught by each of Hoff and Berg. Pet. 24–25. Petitioner argues that Hoff illustrates a client device with processors that execute computer program modules like the tuner and channel applications.

Id. (citing Ex. 1004, Fig. 1B, elements 143, 152, 153; Madisetti Decl. ¶ 119). Petitioner also contends that Berg discloses that a tuner initializes a channel, which causes the GUI to be generated and a webpage to be retrieved and displayed. *Id.* at 25 (citing Ex. 1009, 4).

e) Limitations 1.D and 1.E

Claim limitation 1.D recites that accessing the networked information monitor defined by the networked information monitor template results in "transmission, over a network to a web server at a network location, of a content request for content to be displayed within the frame of the viewer graphical user interface defined by the networked information monitor template." Limitation 1.E recites that accessing the networked information monitor further results in "reception, over the network from the web server at the network location, of content transmitted from the web server in response to the content request, the content being time-varying." Petitioner contends that Berg discloses limitation 1.D when it describes that the initiation of the exemplary channel causes a request to fetch a web page over the Internet. Pet. 25–26 (citing Ex. 1009, 3–4; Madisetti Decl. ¶¶ 123–124). Petitioner further argues that Berg describes that, in response to this request, web content is received over the network, and that this, in combination with the teachings of Nazem or APA relating to time-varying content, teaches limitation 1.E. *Id.* at 26 (citing Ex. 1009, 3, 4, 7; Ex. 1007, 1:60–2:14; Ex. 1001, 1:56–67; Madisetti Decl. ¶¶ 125–128).

f) Limitations 1.F and 1.G

Claim limitation 1.F recites that accessing the networked information monitor defined by the networked information monitor template results in "presentation, on the display, of the viewer graphical user interface defined by the networked information monitor template outside of and separate from

any graphical user interface of any other application." Claim limitation 1.G adds that accessing the networked information monitor defined by the networked information monitor template results in "presentation, on the display within the frame of the viewer graphical user interface defined by the networked information monitor, of the time-varying content received from the web server."

Petitioner contends that Berg discloses these limitations (with the exception of the time-varying nature of the content) in its description of the channel opening a window on a screen and displaying webpage content within that window. Pet. 26–27 (citing Ex. 1009, 2 ("Channels construct their own frames if and when they need them."), 3, 4 ("When the user makes a selection [of a URL corresponding to a web location with content], the program fetches that content and displays it in a web viewing window"), 7; Madisetti Decl. ¶¶ 131, 133–136). As before, Petitioner contends that Nazem or APA discloses the time-varying nature of some web content. *Id.* at 27 (citing Ex. 1007, 1:60–2:14; Ex. 1001, 1:56–67; Madisetti Decl. ¶ 135).

g) Combination of Prior Art References

Even if Petitioner's prior art references disclose all of the limitations in claim 1 when combined, there must be evidence to explain why a person of ordinary skill in the art would have combined the references to arrive at the claimed invention. *Kinetic Concepts, Inc. v. Smith & Nephew, Inc.*, 688 F.3d 1342, 1366–67 (Fed. Cir. 2012) (citing *Innogenetics, N.V. v. Abbott Labs.*, 512 F.3d 1363, 1374 (Fed. Cir. 2008) (holding that post-*KSR* "some kind of motivation must be shown from some source, so that the [trier of fact] can understand why a person of ordinary skill would have thought of either combining two or more references or modifying one to achieve the

patented [invention]")). A precise teaching directed to the specific subject matter of a challenged claim is not necessary to establish obviousness. *KSR*, 550 U.S. at 418. Rather, "any need or problem known in the field of endeavor at the time of invention and addressed by the patent can provide a reason for combining the elements in the manner claimed." *Id.* at 420. Accordingly, Petitioner must show that "a skilled artisan would have been motivated to combine the teachings of the prior art references to achieve the claimed invention, and that the skilled artisan would have had a reasonable expectation of success in doing so." *In re Magnum Oil Tools Int'l, Ltd.*, 829 F.3d 1364, 1381 (Fed. Cir. 2016) (internal quotations and citations omitted).

Here, Petitioner asserts that a person of ordinary skill in the art would have been motivated to combine the teachings of Hoff and Berg because they "disclose similar systems in which a client with a tuner downloads channels from a transmitter on a server over the Internet." Pet. 35 (citing Madisetti Decl. ¶¶ 217–21). Petitioner explains that "*Hoff* discloses the channel application can be any 'software application'" (*id.* (citing Ex. 1004, Abstract, 4:7–10), and "*Berg* discloses the channel applications can be Java software applications and provides an example of one such Java application" (*id.* (citing Ex. 1009, 1–4)). Petitioner reasons that it would have been obvious to a person of ordinary skill in the art "to employ the system described in *Hoff* to download and run the channel disclosed in *Berg*, as *Berg* indicates its channel can be implemented on such a system," and "[b]ecause *Berg* expressly indicates its channel is designed for a system like the one in *Hoff*, a [person of ordinary skill in the art] would have had a reasonable expectation that the channel application in *Berg* could have been successfully run on the *Hoff* system . . . ." *Id.* at 36–37 (citing Madisetti Decl. ¶ 221). Petitioner additionally argues that one of ordinary skill would

have been motivated to modify the Berg channel to include a URL for a web page with time-varying content to provide a user with up-to-date information as opposed to static information. *Id.* at 36 (citing Madisetti Decl. ¶ 220). On the current record, Petitioner's argument is reasonable in light of its cited evidence.

h) Conclusion

Based on the present record, we find that Petitioner has shown a reasonable likelihood of prevailing with respect to its challenge of claim 1 as unpatentable as being obvious over Hoff, Berg, and Nazem or APA, and thus, Petitioner's evidence is sufficient for the purpose of instituting *inter partes* review. Having determined that Petitioner meets the threshold for review of claim 1 based on this ground of obviousness, we institute a review as to all of challenged claims and grounds contained in the Petition. *See SAS*, 138 S. Ct. at 1359–60; *see Consolidated Trial Practice Guide*, 5, 63 (November 2019), https://www.uspto.gov/sites/default/files/documents/tpgnov.pdf ("The Board will not institute on fewer than all claims or all challenges in a petition."). We provide further analysis below in the interest of completeness and to provide guidance to the parties.

6. Claims 9–13, 21–24

Petitioner largely refers to the arguments with respect to claim 1 when discussing how the combination of Hoff, Berg, and Nazem or APA teaches or suggests the limitations in independent claim 13. Pet. 30–32 (citing, for the preamble of claim 13, Ex. 1004, code (57); Ex. 1009, 2; Madisetti Decl. ¶¶ 161–165). Petitioner additionally provides analysis explaining how the combination teaches or suggests the limitations in claims 9–12, which depend from independent claim 1, and claims 21–24, which depend from independent claim 13. Pet. 27–30, 32–35. We note specifically that claim

12, which depends from claim 1, requires the transmission over the network to a server a request for the networked information monitor template, the reception from the server of the template, and the storage of the template in the electronic storage, which Petitioner argues is disclosed in Hoff's and Berg's discussions of a tuner requesting a channel from a transmitter. Pet. 29–30 (citing Ex. 1004, code (57), 2:45–48, 2:56–67, 3:1–5, 3:34–36, 4:24–25, 4:47–57, Fig. 1A; Ex. 1009, 2; Madisetti Decl. ¶¶ 150–160). Claim 24, which depends from claim 13, contains substantially similar limitations to the transmission and reception of claim 12, and is argued on substantially similar basis. *Id.* at 34–35.

Based on our review of Petitioner's arguments and evidence concerning claims 9–13 and 21–24, we determine that Petitioner has shown a reasonable likelihood of proving that the challenged claims are unpatentable as obvious over Hoff, Berg, and Nazem or APA.

E. Obviousness over Hoff, Berg, Nazem or APA, and Fortin

Petitioner contends that the subject matter of claims 8 and 20 would have been obvious over the combined teachings of Hoff, Berg, Nazem or APA, and Fortin. Pet. 37–41. We have reviewed the information provided by Petitioner, including the relevant portions of the supporting Madisetti Declaration. Based on the current record, for reasons set forth below, we are persuaded that Petitioner has demonstrated a reasonable likelihood of prevailing on this obviousness challenge.

1. Fortin

Fortin is titled "Document Markup Language and System and Method for Generating and Displaying Documents Therein," and published on February 21, 2002, from an application filed on January 23, 1998. Ex. 1008,

codes (54), (45), (22). Petitioner contends Fortin is prior art under 35 U.S.C. § 102(e). Pet. 13.

Fortin describes the use of a markup language to improve the ability of users to view complex documents. Ex. 1008, code (57). Fortin describes how a Java applet can be used by a browser to process documents, provided in a markup language, for display to a user. *Id.* ¶¶ 62–64.

2. Claims 8 and 20

Petitioner provides analysis explaining how the combination of Berg and Fortin teaches the additional limitations of claims 8 (dependent from claim 1) and claim 20 (dependent from claim 13) that "the networked information monitor template includes a markup language file" and in claim 20 that "storing the networked information monitor template comprises storing the markup language file." Pet. 37–40 (citing Ex. 1009, 3, 4, 6; Ex. 1008 ¶¶ 62–64; Madisetti Decl. ¶¶ 222–235). Petitioner additionally explains why one of ordinary skill in the art would have recognized that the functionality described in Fortin could be utilized with the Berg channel. *Id.* at 38–41 (citing Ex. 1009, 3, 4; Ex. 1008 ¶¶ 6–8, 62–65; Madisetti Decl. ¶¶ 226, 236–240).

Based on our review of Petitioner's arguments and evidence concerning claims 8 and 20, we have determined that the Petitioner has shown a reasonable likelihood of proving that the challenged claims are unpatentable as obvious over Hoff, Berg, Nazem or APA, and Fortin.

F. Obviousness over Razavi and Andersen

Petitioner contends that the subject matter of claims 1, 9–13, and 21–24 would have been obvious over the combined teachings of Razavi and Andersen. Pet. 41–64. We have reviewed the information provided by Petitioner, including the relevant portions of the supporting Madisetti

Declaration. Based on the current record, for reasons set forth below, we are persuaded that Petitioner has demonstrated a reasonable likelihood of prevailing on this obviousness challenge.

1. Razavi

Razavi is titled "Detachable Java Applets," and issued on June 4, 2002. Ex. 1006, codes (54), (45). Petitioner contends Razavi is prior art under 35 U.S.C. § 102(e). Pet. 10.

Razavi describes "detaching Java applets from the constraints of the application such as a browser which provides the Java engine for executing those applets." Ex. 1006, code (57). In Razavi, a Java applet will be detachable from the application in which it was spawned and have the functionality of any ordinary application window running in the operating environment, including modifying the interface to suit the peculiarities of the applet. *Id.* at 4:41–56, 5:64–6:20. According to Razavi, the applet must be launched in the virtual machine of the application, and is detached if the user activates a detach user interface element. *Id.* at 5:48–63. Figure 4, reproduced below, is Razavi's illustration of a detached applet as rendered on a display screen. *Id.* at 7:8–9.

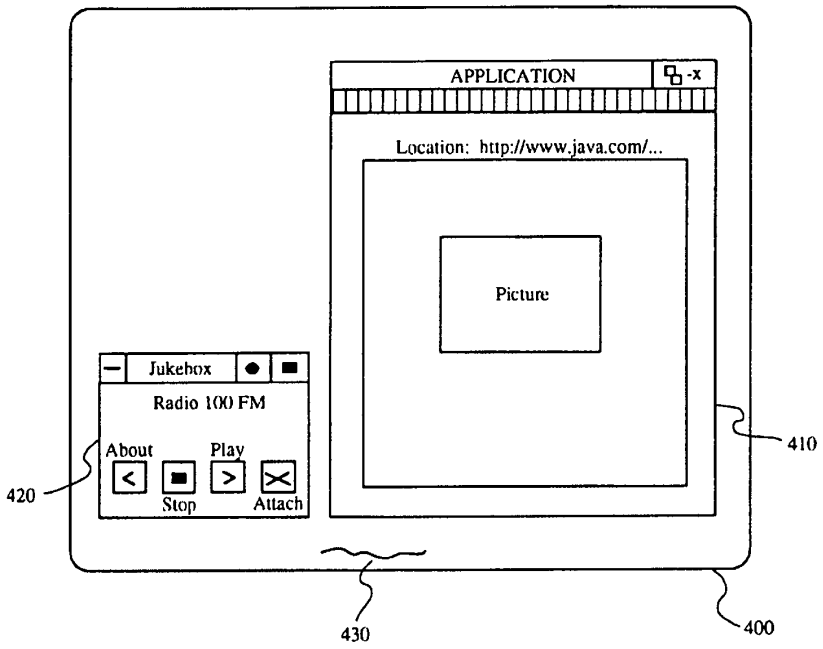


FIGURE 4

Figure 4 depicts application window 410 and detached applet 420 named "Jukebox" displayed on display 400. *Id.* at 7:10–12, 7:29–33. Application window 410 is running over an operating environment desktop 430, which also controls the general look-and-feel of the window for detached applet 420. *Id.* at 7:17–19, 17:39–41. Detached applet 420 contains applet controls "About," "Stop," and "Play" and a control called "Attach," which closes the detached applet 420 and "collapse[s] the applet back into application window 410." *Id.* at 7:34–35, 8:10–14.

The exemplary detached applet described in Razavi, the Jukebox applet, is a streaming audio applet that "creates VCR-type controls that allow the user to stop, play, or rewind the audio stream or to jump to a different selection" and optionally provides an image, such as a CD cover, to

be displayed, associated with each audio selection. *Id.* at 4:25–26, App'x A col. 1.

2. Andersen

Andersen is titled "Database Access Using Active Server Pages," and issued on December 7, 1999. Ex. 1012, codes (54), (45). Petitioner contends Andersen is prior art under 35 U.S.C. § 102(e). Pet. 12. Andersen describes an applet on a client computer that allows a database query to be embedded in an URL in order to obtain query results in HTML form. Ex. 1012, code (57), 3:5–29, 5:3–33. Andersen provides an example of a stock price quoting system implemented using a Java applet. *Id.* at 1:62–63, 1:67–2:30. The applet requests one or more stock prices from a server and the server responds with an HTML file into which data returned by the database is inserted in response to the query. *Id.* at 5:34–7:5. This allows an advantage over a system in which stock prices are available directly from an HTML page rather than from a database, because "storing the stock price data in static HTML pages can be quite inefficient, and data stored in this manner is not amenable to incremental updates and easy and efficient ad hoc searching and manipulation." *Id.* at 2:59–3:4.

3. Analysis of Claim 1

a) Preamble

The preamble of claim 1 recites a "client computing device configured to access content over a network." Petitioner asserts that this preamble is taught by Razavi's and Andersen's disclosure of "similar computer-implemented methods in which an 'applet' is downloaded onto a client computing device, retrieves content from a server over a network, generates a GUI on a display, and presents the content within the GUI." Pet. 41–42 (citing Ex. 1007, 1:1–41; Ex. 1012, 1:10–2:30; Madisetti Decl. ¶ 241).

b) Limitation 1.A

Claim limitation 1.A recites that the client computing device comprises "electronic storage configured to store networked information monitor template associated with a networked information monitor." Petitioner asserts that this limitation is taught by Razavi, in which Razavi discusses that an applet can be downloaded automatically and launched in a virtual machine at the client site when a user visits a web site which includes a statement to launch the applet. Pet. 43–44 (citing, *inter alia*, Ex. 1006, 5:51–56). Petitioner asserts that one of ordinary skill would have understood that when such applets are downloaded and executed, they would be stored in electronic storage of the client computing device. Pet. 44 (citing Madisetti Decl. ¶¶ 247–248).

c) Limitation 1.B

Claim limitation 1.B recites that "the networked information monitor template having therein a definition of a viewer graphical user interface having a frame within which time-varying content in a web browser-readable language may be presented on a display associated with the client computing device, wherein the frame of the viewer graphical user interface lacks controls for enabling a user to specify a network location at which content for the networked information monitor is available." Petitioner asserts that this limitation is taught by Razavi or the combination of Razavi and Andersen. Pet. 44–47. Petitioner argues that Razavi's Jukebox applet defines the GUI generated by the applet and opens audio content from a server as specified via an HTML tag. *Id.* at 44–46 (citing Ex. 1006, App'x A cols. 1–4, 7; Madisetti Decl. ¶¶ 249–250). Petitioner notes that the Jukebox applet retrieves not only streaming audio files, but also retrieves

images and titles associated with the audio files, to be displayed. *Id.* at 45 (citing Ex. 1006, App'x A col. 1).

Petitioner alternatively argues that Andersen's stock price applet presents time-varying content in a web-browser readable language, because an HTML file is received for specified stocks and the content is presented to the user on the display. Pet. 46 (citing Ex. 1012, 6:34–40; Madisetti Decl. ¶ 252). While Andersen's stock prices are displayed within a browser, Petitioner argues that one of ordinary skill would have found it obvious to detach the applet from the browser as taught by Razavi. *Id.* at 46–47 (citing Madisetti Decl. ¶ 255).

Petitioner asserts that both Razavi's and Andersen's applets lack control for a user to specify the network location from which the content to be displayed is available. *Id.* at 46 (citing Madisetti Decl. ¶ 254).

d) Limitation 1.C

Claim limitation 1.C recites that "one or more processors configured to execute one or more computer program modules, the one or more computer program modules being configured to access the networked information monitor defined by the networked information monitor template." Petitioner asserts that this limitation is taught by Razavi or the combination of Razavi and Andersen. Pet. 47–48. Petitioner argues one of ordinary skill would have understood that Razavi's operating and desktop environments would reside on a client device including processors. *Id.* at 47 (citing Ex. 1006, 3:47–65; Madisetti Decl. ¶ 259). Petitioner also argues that it was well known to use a computing device with processors to execute applications such as the applet of Andersen. *Id.* at 47–48 (citing Ex. 1012, 4:25–53; Madisetti Decl. ¶ 260).

e) Limitations 1.D and 1.E

Claim limitation 1.D recites that accessing the networked information monitor defined by the networked information monitor template results in "transmission, over a network to a web server at a network location, of a content request for content to be displayed within the frame of the viewer graphical user interface defined by the networked information monitor template." Limitation 1.E recites that accessing the networked information monitor further results in "reception, over the network from the web server at the network location, of content transmitted from the web server in response to the content request, the content being time-varying." Petitioner asserts that these limitation are taught by Razavi or the combination of Razavi and Andersen. Pet. 48–50. Petitioner contends that Razavi discloses a request for and receipt of networked information, including audio songs, titles, or images, from a server. *Id.* at 48–49 (citing Ex. 1006, App'x A cols. 1–4; Madisetti Decl. ¶¶ 262, 265). Petitioner further argues that Andersen's Java applet requests and receives time-varying content for display, namely the stock prices received from a URL associated with a server system. *Id.* (citing Ex. 1012 4:42–53, 6:34–40, Fig. 2; Madisetti Decl. ¶¶ 263, 266).

f) Limitations 1.F and 1.G

Claim limitation 1.F recites that accessing the networked information monitor defined by the networked information monitor template results in "presentation, on the display, of the viewer graphical user interface defined by the networked information monitor template outside of and separate from any graphical user interface of any other application." Claim limitation 1.G adds that accessing the networked information monitor defined by the networked information monitor template results in "presentation, on the display within the frame of the viewer graphical user interface defined by

the networked information monitor, of the time-varying content received from the web server."

Petitioner contends that Razavi discloses these limitations, and that alternatively, the combination of Razavi and Andersen discloses limitation 1.G. Pet. 50–54. Petitioner argues that Razavi's disclosure of the detached state of the Jukebox applet is presented outside and separate from the application that launched the applet. *Id.* at 50–52 (citing Ex. 1006, 4:3–9, 5:64–6:20, 7:8–33, 8:27–29, Fig. 4, App'x A col. 7; Madisetti Decl. ¶¶ 268–271). Petitioner refers to the presentation of songs, titles, and images in Razavi as teaching the presentation of time-varying content from the web server within the frame of the graphical user interface. *Id.* at 52–53 (citing Ex. 1006, 1:58–62, 4:25–26, Fig. 4, App'x A cols. 1–2.; Madisetti Decl. ¶ 272). Petitioner again argues that the Andersen stock prices constitute time-varying data retrieved from a server and displayed for a user. *Id.* at 53–54 (citing Ex. 1012, 6:34–40, Fig. 2).

g) Combination of Prior Art References

Petitioner asserts that a person of ordinary skill in the art would have been motivated to combine the teachings of Razavi and Andersen because they "disclose similar systems in which Java applets can be downloaded over the Internet from a server and executed on a client device." Pet. 62. Petitioner argues that one of ordinary skill in the art would have been motivated to and reasonably expected to be successful in a combination, as Andersen's client device and network architecture were conventional for use with programs such as Razavi's applet. *Id.* at 62–64 (citing Madisetti Decl. ¶¶ 362–363). On the current record, Petitioner's argument is reasonable in light of its cited evidence.

h) Conclusion

Based on the present record, we find that Petitioner has shown a reasonable likelihood of prevailing with respect to its challenge of claim 1 as unpatentable as being obvious over Razavi and Andersen.

4. Claims 9–13, 21–24

Petitioner largely refers to the arguments with respect to claim 1 when discussing how the combination of Razavi and Andersen teaches or suggests the limitations in independent claim 13. Pet. 58–60 (citing, for the preamble of claim 13, Ex. 1006, 1:1–41, App'x A; Ex. 1012, 1:10–2:30; Madisetti Decl. ¶¶ 302–306). Petitioner additionally provides analysis explaining how the combination teaches or suggests the limitations in claims 9–12, which depend from independent claim 1, and claims 21–24, which depend from independent claim 13. Pet. 54–58, 60–62. Petitioner argues that claim 12's additional limitations are disclosed in Razavi's discussion of the applet as downloaded and received pre-compiled and launched automatically. Pet. 57–58 (citing Ex. 1007, 1:9–25, 5:51–56, 7:29–33; Madisetti Decl. ¶¶ 293–294, 297–301). Claim 24, which depends from claim 13, contains substantially similar limitations to the transmission and reception of claim 12, and is argued on a substantially similar basis. *Id.* at 61–62.

Based on our review of Petitioner's arguments and evidence concerning claims 9–13 and 21–24, we determine that Petitioner has shown a reasonable likelihood of proving that the challenged claims are unpatentable as obvious over Razavi and Andersen.

G. Obviousness over Razavi, Andersen, and Fortin

Petitioner contends that the subject matter of claims 8 and 20 would have been obvious over the combined teachings of Razavi, Andersen, and Fortin. Pet. 64–67. We have reviewed the information provided by

Petitioner, including the relevant portions of the supporting Madisetti Declaration. Based on the current record, for reasons set forth below, we are persuaded that Petitioner has demonstrated a reasonable likelihood of prevailing on this obviousness challenge.

Petitioner provides analysis explaining how the combination of Razavi and Andersen teaches the additional limitations of claims 8 (dependent from claim 1) and claim 20 (dependent from claim 13) that "the networked information monitor template includes a markup language file" and in claim 20 that "storing the networked information monitor template comprises storing the markup language file." Pet. 64–65 (citing Ex. 1007, 7:34–35, App'x A col. 1; Madisetti Decl. ¶¶ 365–370). Petitioner additionally explains why one of ordinary skill in the art would have recognized that the functionality described in Fortin could be utilized with the applets taught in Razavi and Andersen, in order to provide more advanced GUI controls such as dynamic menus. *Id.* (citing Ex. 1008 ¶¶ 6, 62, 64, 65, 67, Fig. 8; Madisetti Decl. ¶¶ 368, 369, 376).

Based on our review of Petitioner's arguments and evidence concerning claims 8 and 20, we determine that Petitioner has shown a reasonable likelihood of proving that the challenged claims are unpatentable as obvious over Razavi, Andersen, and Fortin.

III. CONCLUSION

For the reasons expressed above, we determine that Petitioner has established the requisite reasonable likelihood of prevailing as to at least one challenged claim of the '407 patent.

Our factual findings and determinations at this stage of the proceeding are preliminary, and based on the evidentiary record developed thus far. This is not a final decision as to the patentability of claims for which *inter*

partes review is instituted. Our final decision will be based on the record as fully developed during trial.

IV. ORDER

It is, therefore,

ORDERED that, pursuant to 35 U.S.C. § 314(a), an *inter partes* review of claims 1, 8–13, and 20–24 of the '407 patent with respect to all grounds in the Petition is hereby instituted; and

FURTHER ORDERED that pursuant to 35 U.S.C. § 314(c) and 37 C.F.R. § 42.4, notice is hereby given of the institution of a trial commencing on the entry date of this Decision.

IPR2019-01279
Patent 8,510,407 B1

FOR PETITIONER:

Charanjit Brahma
Troutman Sanders LLP
Charanjit.brahma@troutmansanders.com

Dustin Weeks
Troutman Sanders LLP
Dustin.weeks@troutmansanders.com

PATENT OWNER:

Lewis E. Hudnell, III
Hudnell Law Group P.C.
lewis@hudnelllaw.com

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	ISSUE DATE	PATENT NO.	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/932,553	08/13/2013	8510407	043311-0381668	2604

909 7590 07/24/2013

Pillsbury Winthrop Shaw Pittman, LLP (NV)
PO Box 10500
McLean, VA 22102

ISSUE NOTIFICATION

The projected patent number and issue date are specified above.

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)
(application filed on or after May 29, 2000)

The Patent Term Adjustment is 928 day(s). Any patent to issue from the above-identified application will include an indication of the adjustment on the front page.

If a Continued Prosecution Application (CPA) was filed in the above-identified application, the filing date that determines Patent Term Adjustment is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) WEB site (http://pair.uspto.gov).

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Application Assistance Unit (AAU) of the Office of Data Management (ODM) at (571)-272-4200.

APPLICANT(s) (Please see PAIR WEB site http://pair.uspto.gov for additional applicants):

John Albert Kembel, Palo Alto, CA;
George Andrew Kembel, Menlo Park, CA;
Daniel S. Kim, Palo Alto, CA;
John Russell, Palo Alto, CA;
Jake Wobbrock, Palo Alto, CA;
Geoffrey S. Kembel, Menlo Park, CA;
Jeremy L. Kembel, Palo Alto, CA;
Lynn D. Gabbay, Sunnyvale, CA;

The United States represents the largest, most dynamic marketplace in the world and is an unparalleled location for business investment, innovation, and commercialization of new technologies. The USA offers tremendous resources and advantages for those who invest and manufacture goods here. Through SelectUSA, our nation works to encourage and facilitate business investment. To learn more about why the USA is the best country in the world to develop technology, manufacture products, and grow your business, visit SelectUSA.gov.

SUBSTITUTE SPECIFICATION (CLEAN)

DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] The present application is a continuation of and incorporates by reference U.S. Non-Provisional patent application Ser. No. 09/558,925, filed Apr. 26, 2000, now U.S. Patent No. 7,660,868, which claims priority from and incorporates by reference U.S. Provisional Application ser. nos. 60/131,083, filed Apr. 26, 1999, 60/131,114, filed Apr. 26, 1999, 60/131,115, filed Apr. 26, 1999, 60/176,687, filed Jan. 18, 2000, and 60/176,699, filed Jan. 18, 2000. The present application claims priority to U.S. Non-Provisional patent application ser. no. 09/558,925, filed Apr. 26, 2000, now U.S. Patent No. 7,660,868, and each of the aforementioned applications to which it claims priority.

[0002] The present application is also related to and incorporates by reference the following U.S. patent applications: Non-Provisional Application ser. no. 09/558,922, filed April 26, 2000, now U.S. Patent No. 7,756,967; Non-Provisional Application ser. no. 09/558,923, filed April 26, 2000; Non-Provisional Application ser. no. 09/558,924, filed April 26, 2000, now U.S. Patent No. 7,356,569; Non-Provisional Application Ser. No. ~~xx/xxx,xxx~~ 11/932,286, filed October 31, 2007 titled "Component For Accessing And Displaying Internet Content" (attorney docket no. IMS 07-05-01); Non-Provisional Application ser. no. 11/932,286, filed Oct. 31, 2007, titled "Server Including Components For Accessing And Displaying Internet Content And For Providing Same To A Client"; Non-Provisional Application ser. no.

Change(s) applied to document, /L.K.H./ 4/29/2013

Receipt date: 02/25/2008	Application Number	11932553	11932553 - GAU: 2446
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Filing Date	2007-10-31	
(Not for submission under 37 CFR 1.99)	First Named Inventor	John Albert Kembel	
	Art Unit	N/A	
	Examiner Name	Not yet assigned	
	Attorney Docket Number	IMS 07-05-06	

	42	6538673		2003 ~~2005~~-03-03	Maslov	
Change(s) applied to document, /S.R.R./ 5/15/2013	43	6549612		2003-04-15	Gifford et al.	
	44	6560639		2003-05-06	Dan et al.	
	45	6594682		2003 ~~2005~~-07-03	Peterson et al.	
	46	6369840		2002-04-09	Barnett et al.	
	47	6751606		2004-06-15	Fries et al.	
	48	6784900		2004 ~~2005~~-08-04	Dobronsky et al.	
	49	6816880		2004-11-09	Strandberg et al.	
	50	6842779		2005-01-11	Nishizawa	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear

EFS Web 2.0.2

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /B.R.B

Receipt date: 02/25/2008	Application Number	11932553	11932553 - GAU: 2446
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Filing Date	2007-10-31	
(Not for submission under 37 CFR 1.99)	First Named Inventor	John Albert Kembel	
	Art Unit	N/A	
	Examiner Name	Not yet assigned	
	Attorney Docket Number	IMS 07-05-06	

	31	6297819		2001 ~~2005~~-10-01	Furst	
Change(s) applied to document, /S.R.R./ 5/15/2013	32	6314451		2001-11-06	Landsman et al.	
	33	6341305		2002-01-22	Wolfe	
	34	6393407		2002-05-21	Middleton et al.	
	35	6411992		2002-06-25	Srinivasan et al.	
	36	6418440		2002-07-09	Kuo et al.	
	37	6434563		2002 ~~2005~~-08-02	Pasquali et al.	
	38	6460029		2002 ~~2005~~-10-02	Fries et al.	
	39	6484149		2002-11-19	Jammes et al	
	40	6487566		2002 ~~2005~~-11-02	Sundaresan	
	41	6537324		2003-03-25	Tabata et al.	

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /B.R.B

Receipt date: 02/25/2008	Application Number	11932553	11932553 - GAU: 2446
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Filing Date	2007-10-31	
	First Named Inventor	John Albert Kembel	
	Art Unit	N/A	
(Not for submission under 37 CFR 1.99)	Examiner Name	Not yet assigned	
	Attorney Docket Number	IMS 07-05-06	

	20	6018344		2000-01-25	Harada et el.	
	21	6026433		2000-02-15	D'Arlach et al.	
	22	6065044		2000-05-16	Ogasawara	
	23	6133916		2000-10-01	Bukszar et al.	
	24	6161112		2000-12-12	Cragun et al.	
	25	6177936		2001-01-23	Cragun	
	26	6199082		2001 ~~2005~~-03-01	Ferrel et al.	
Change(s) applied to document. /S.R.R./ 5/15/2013	27	6230173		2001 ~~2005~~-05-01	Ferrel et al.	
	28	6237030		2001-05-22	Adams et al.	
	29	6268856		2001-07-31	Bruck et al.	
	30	6275854		2001-08-14	Himmel et al	

EFS Web 2.0.2

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /B.R.B

Receipt date: 02/25/2008	Application Number	11932553 11932553 - GAU: 2446
INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Filing Date	2007-10-31
	First Named Inventor	John Albert Kembel
(Not for submission under 37 CFR 1.99)	Art Unit	N/A
	Examiner Name	Not yet assigned
	Attorney Docket Number	IMS 07-05-06

	9	5809248		1998-09-15	Vidovic	
	10	5890172		1999-03-01	Borman et al	
Change(s) applied to document. /S.R.R./ 5/15/2013	11	5893091		1999 ~~1996~~-04-06	Hunt et al	
	12	5948061		1999-09-07	Merriman et al	
	13	5974446		1999-10-26	Sonnenreich et al	
	14	5977964		1999-11-01	Williams et al.	
	15	5983227		1999-11-09	Nazem et al.	
	16	5987513		1999-11-16	Prithviraj et al.	
	17	5995756		1999-11-01	Hermann	
	18	6012090		2000-01-04	Chung et al.	
	19	6012098		2000-01-01	Bayeh et al.	

EFS Web 2.0.2

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /B.R.B

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/932,553	10/31/2007	John Albert Kembel	043311-0381668	2604

909 7590 07/15/2013

Pillsbury Winthrop Shaw Pittman, LLP (NV)
PO Box 10500
McLean, VA 22102

EXAMINER	
LIN, KENNY S	
ART UNIT	PAPER NUMBER
2478	

NOTIFICATION DATE	DELIVERY MODE
07/15/2013	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

docket_ip@pillsburylaw.com
margaret.drosos@pillsburylaw.com

UNITED STATES DEPARTMENT OF COMMERCE
U.S. Patent and Trademark Office
Address : COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450

APPLICATION NO./ CONTROL NO.	FILING DATE	FIRST NAMED INVENTOR / PATENT IN REEXAMINATION	ATTORNEY DOCKET NO.
11/932,553	31 October, 2007	KEMBEL ET AL.	043311-0381668

Pillsbury Winthrop Shaw Pittman, LLP (NV)
PO Box 10500
McLean, VA 22102

EXAMINER	
KENNY LIN	
ART UNIT	PAPER
2478	20130710

DATE MAILED:

Please find below and/or attached an Office communication concerning this application or proceeding.

Commissioner for Patents

7/1/13 IDS considered

/Kenny S Lin/
Primary Examiner, Art Unit 2478

PTO-90C (Rev.04-03)

INFORMATION DISCLOSURE CITATION IN AN APPLICATION	Docket Number	Application Number
	043311-0381668	11/932,553
	Applicant(s)	
	John Albert KEMBEL *et al.*	
	Filing Date	Group Art Unit
Form PTO-1449	October 31, 2007	2478

U.S. PATENT DOCUMENTS

*EXAMINER INITIAL	REF	DOCUMENT NO.	DATE	NAME	CLASS	SUBCLASS	FILING DATE IF APPROPRIATE
	A	6,023,698	02/08/2000	Lavey, Jr., *et al.*	707	10	
	B	6,031,904	02/29/2000	An et al.	379	201	
	C	6,044,403	03/28/2000	Gerszberg et al.	709	225	
	D	6,216,141	04/10/2001	Straub et al.	707	513	
	E	6,278,448	08/21/2001	Brown et al.	345	333	
	F	6,343,377	01/29/2002	Gessner *et al.*	717	10	
	G	6,629,143	09/30/2003	Pang	709	226	
	H	6,819,345	11/16/2004	Jones et al.	345	856	
	I	2003/0051027	03/13/2003	Aupperle et al.	709	224	

FOREIGN PATENT DOCUMENTS

	REF	DOCUMENT NO.	DATE	COUNTRY	CLASS	SUBCLASS	Translations YES	Translations NO

OTHER DOCUMENTS *(Including Author, Title, Date, Pertinent Pages, Etc.)*

	J	WHITEHEAD et al., "WEBDAV: IETF Standard for Collaborative Authoring on the Web", September and October 1998, Retrieved from the Internet <URL:ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=722228>, pages 1-7 as printed.

EXAMINER	DATE CONSIDERED
/Kenny Lin/	07/10/2013

EXAMINER: Initial if citation considered, whether or not citation is in conformance with MPEP Section 609; Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to Applicant.

404021132v1

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /KL/

OPAP IAP14 JUL 03 2013 PATENT & TRADEMARK OFFICE

IFW

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), to: Mail Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450
or Fax (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

909 7590 04/18/2013
Pillsbury Winthrop Shaw Pittman, LLP (NV)
PO Box 10500
McLean, VA 22102

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission
I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being facsimile transmitted to the USPTO (571) 273-2885, on the date indicated below.

HANNAH TRAN	(Depositor's name)
[signature]	(Signature)
7/3/13	(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/932,553	10/31/2007	John Albert Kembel	043311-0381668	2604

TITLE OF INVENTION: DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$890	$0	$0	$890	07/18/2013

EXAMINER	ART UNIT	CLASS-SUBCLASS
LIN, KENNY S	2478	709-217000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).
☐ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.
☐ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-02 or more recent) attached. Use of a Customer Number is required.

2. For printing on the patent front page, list
(1) the names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.
1 PILSBURY WINTHROP SHAW PITTMAN, LLP
2
3

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)
PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document has been filed for recordation as set forth in 37 CFR 3.11. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE: MAINSTREAM SCIENTIFIC, LLC
(B) RESIDENCE: (CITY and STATE OR COUNTRY): MOUNTAIN VIEW, CA

Please check the appropriate assignee category or categories (will not be printed on the patent): ☐ Individual ☑ Corporation or other private group entity ☐ Government

4a. The following fee(s) are submitted:
☑ Issue Fee
☐ Publication Fee (No small entity discount permitted)
☐ Advance Order - # of Copies _____

4b. Payment of Fee(s): (Please first reapply any previously paid issue fee shown above)
☐ A check is enclosed.
☑ Payment by credit card. Form PTO-2038 is attached.
☐ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment, to Deposit Account Number _____ (enclose an extra copy of this form).

~~07/05/2013 MBELETE2 00000041 11932553~~
~~01 FC:2501 890.00 OP~~

07/05/2013 MBELETE2 00000042 11932553
01 FC:2501 890.00 OP

PAGE 2/4 * RCVD AT 7/3/2013 7:30:18 PM [Eastern Daylight Time] * SVR:W-PTOFAX-002/39 * DNIS:2732885 * CSID: * DURATION (mm-ss):01-34

5. Change in Entity Status (from status indicated above)

☐ Applicant certifying micro entity status. See 37 CFR 1.29 — NOTE: Absent a valid certification of Micro Entity Status (see form PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.

☐ Applicant asserting small entity status. See 37 CFR 1.27 — NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.

☐ Applicant changing to regular undiscounted fee status. — NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: The Issue Fee and Publication Fee (if required) will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the United States Patent and Trademark Office.

Authorized Signature Rakesh Ramde — Date July 3, 2013

Typed or printed name RAKESH RAMDE — Registration No. _____

This collection of information is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, Virginia 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450.

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PAGE 4/4 * RCVD AT 7/3/2013 7:30:18 PM [Eastern Daylight Time] * SVR:W-PTOFAX-002/39 * DNIS:2732885 * CSID: * DURATION (mm-ss):01-34

OPAP JUL 03 2013 PATENT & TRADEMARK OFFICE

Doc Code: TRAN.LET
Document Description: Transmittal Letter

PTO/SB/21 (07-09)
Approved for use through 07/31/2012. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Total Number of Pages in This Submission: 4

Application Number	~~11/932,553~~ 11/932,553
Filing Date	10/31/2007
First Named Inventor	John Albert Kembel
Art Unit	2478
Examiner Name	Lin, Kenny S
Attorney Docket Number	043311-0381688

ENCLOSURES *(Check all that apply)*

- [x] Fee Transmittal Form
  - [x] Fee Attached
- [ ] Amendment/Reply
  - [ ] After Final
  - [ ] Affidavits/declaration(s)
- [ ] Extension of Time Request
- [ ] Express Abandonment Request
- [ ] Information Disclosure Statement
- [ ] Certified Copy of Priority Document(s)
- [ ] Reply to Missing Parts/ Incomplete Application
  - [ ] Reply to Missing Parts under 37 CFR 1.52 or 1.53
- [ ] Drawing(s)
- [ ] Licensing-related Papers
- [ ] Petition
- [ ] Petition to Convert to a Provisional Application
- [ ] Power of Attorney, Revocation Change of Correspondence Address
- [ ] Terminal Disclaimer
- [ ] Request for Refund
- [ ] CD, Number of CD(s) _____
  - [ ] Landscape Table on CD
- [ ] After Allowance Communication to TC
- [ ] Appeal Communication to Board of Appeals and Interferences
- [ ] Appeal Communication to TC (Appeal Notice, Brief, Reply Brief)
- [ ] Proprietary Information
- [ ] Status Letter
- [x] Other Enclosure(s) (please Identify below): PTO-2038

Remarks

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm Name	Mainstream Scientific, LLC		
Signature	Rakesh Ramde		
Printed name	Rakesh Ramde		
Date	July 3, 2013	Reg. No.	

CERTIFICATE OF TRANSMISSION/MAILING

I hereby certify that this correspondence is being facsimile transmitted to the USPTO or deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on the date shown below:

Signature	Htran		
Typed or printed name	Hannah Tran	Date	7/3/13

This collection of information is required by 37 CFR 1.5. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and1.14. This collection is estimated to 2 hours to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

INFORMATION DISCLOSURE CITATION IN AN APPLICATION	Docket Number	Application Number
	043311-0381668	11/932,553
	Applicant(s)	
	John Albert KEMBEL *et al.*	
	Filing Date	Group Art Unit
Form PTO-1449	October 31, 2007	2478

U.S. PATENT DOCUMENTS

*EXAMINER INITIAL	REF	DOCUMENT NO.	DATE	NAME	CLASS	SUBCLASS	FILING DATE IF APPROPRIATE
	A	6,023,698	02/08/2000	Lavey, Jr., *et al.*	707	10	
	B	6,031,904	02/29/2000	An et al.	379	201	
	C	6,044,403	03/28/2000	Gerszberg et al.	709	225	
	D	6,216,141	04/10/2001	Straub et al.	707	513	
	E	6,278,448	08/21/2001	Brown et al.	345	333	
	F	6,343,377	01/29/2002	Gessner *et al.*	717	10	
	G	6,629,143	09/30/2003	Pang	709	226	
	H	6,819,345	11/16/2004	Jones et al.	345	856	
	I	2003/0051027	03/13/2003	Aupperle et al.	709	224	

FOREIGN PATENT DOCUMENTS

	REF	DOCUMENT NO.	DATE	COUNTRY	CLASS	SUBCLASS	Translations YES	Translations NO

OTHER DOCUMENTS *(Including Author, Title, Date, Pertinent Pages, Etc.)*

	J	WHITEHEAD et al., "WEBDAV: IETF Standard for Collaborative Authoring on the Web", September and October 1998, Retrieved from the Internet <URL:ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=722228>, pages 1-7 as printed.

EXAMINER	DATE CONSIDERED

EXAMINER: Initial if citation considered, whether or not citation is in conformance with MPEP Section 609; Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to Applicant.

404021132v1

Electronic Patent Application Fee Transmittal	
Application Number:	11932553
Filing Date:	31-Oct-2007
Title of Invention:	DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES
First Named Inventor/Applicant Name:	John Albert Kembel
Filer:	D. Benjamin Esplin/Susan Trader
Attorney Docket Number:	043311-0381668
Filed as Small Entity	

Utility under 35 USC 111(a) Filing Fees

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Miscellaneous:				
Submission- Information Disclosure Stmt	2806	1	90	90
		Total in USD ($)		**90**

Electronic Acknowledgement Receipt

EFS ID:	16206984
Application Number:	11932553
International Application Number:	
Confirmation Number:	2604
Title of Invention:	DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES
First Named Inventor/Applicant Name:	John Albert Kembel
Customer Number:	909
Filer:	D. Benjamin Esplin/Susan Trader
Filer Authorized By:	D. Benjamin Esplin
Attorney Docket Number:	043311-0381668
Receipt Date:	01-JUL-2013
Filing Date:	31-OCT-2007
Time Stamp:	16:29:30
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	Deposit Account
Payment was successfully received in RAM	$90
RAM confirmation Number	3902
Deposit Account	033975
Authorized User	

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)

1		0381668_SIDS.pdf	483398 137e1753750024c01fd2c7547a19261546ad3c4f	yes	4
	Multipart Description/PDF files in .zip description				
	Document Description		**Start**	**End**	
	Transmittal Letter		1	3	
	Information Disclosure Statement (IDS) Form (SB08)		4	4	
Warnings:					
Information:					
2	Non Patent Literature	WHITEHEAD.pdf	1355510 8a30d570cfecfb399bd2c76134a0ddae140aed6a	no	7
Warnings:					
Information:					
3	Fee Worksheet (SB06)	fee-info.pdf	30618 f877c1cc89bed103f91c4ce2e7ac840ade620bcf	no	2
Warnings:					
Information:					
		Total Files Size (in bytes):	1869526		

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICANTS :	John Albert KEMBEL *et al.*	CONFIRMATION NO.:	2604
SERIAL NUMBER :	11/932,553	EXAMINER:	Kenny S. Lin
FILING DATE :	October 31, 2007	ART UNIT:	2478
FOR :	DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES		

SUPPLEMENTAL INFORMATION DISCLOSURE STATEMENT

Commissioner for Patents
P. O. Box 1450
Alexandria, VA 22313-1450

Sir:

Pursuant to the duty of disclosure under 37 C.F.R. §§ 1.56, 1.97 and 1.98, Applicants hereby make of record in the above-identified application the documents listed on the attached Form PTO-1449.

Applicants have listed a publication date on the attached PTO-1449 based on information presently available to the undersigned. However, the listed publication date should not be construed as an admission that the information was actually published on the date indicated.

Applicants reserve the right to establish the patentability of the claimed invention over any of the information provided herewith, and/or to prove that this information may not be prior art, and/or to prove that this information may not be enabling for the teachings purportedly offered.

This Supplemental Information Disclosure Statement is being filed more than three months after the filing date of a national application other than a continued prosecution application under 37 C.F.R. § 1.53(d), or more than three months after the date of entry of the national stage as set forth in 37 C.F.R. § 1.491 in an international application, and after the mailing date of any of a Final Action under 37 C.F.R. § 1.113, a Notice of Allowance under 37 C.F.R. § 1.311, or an action that otherwise closes prosecution in the application, but on or before payment of the Issue Fee. 37 C.F.R. § 1.97(d).

Document I was cited by the U.S. Patent Office in co-pending Application Serial No. 11/932,286, filed October 31, 2007, in a Notice of Allowability dated April 12, 2013. Documents D, E, H, and J were cited by the U.S. Patent Office in co-pending Application Serial No. 11/932,630, filed October 31, 2007, in an Office Action dated April 23, 2013. Documents A and F were cited by the U.S. Patent Office in co-pending Application Serial No. 13/215,874, filed August 23, 2011, in an Office Action dated April 23, 2013. Documents B, C, and G were cited by the U.S. Patent Office in co-pending Application Serial No. 12/702,156, filed February 8, 2010, in an Office Action dated May 23, 2013. I hereby certify that no item of information contained in this Supplemental Information Disclosure Statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to my knowledge after making reasonable inquiry, no item of information contained in this Supplemental Information Disclosure Statement was known to any individual designated in 37 C.F.R. § 1.56(c)

more than three months prior to the filing of this Supplemental Information Disclosure Statement. 37 C.F.R. § 1.97(e)(2).

Please charge the $90.00 fee [37 C.F.R. § 1.17(p)] to Deposit Account No. 03-3975. In addition, please charge any fee deficiencies, or credit any overpayment, to Deposit Account No. 03-3975 as needed to ensure consideration of the Supplemental Information Disclosure Statement.

It is respectfully requested that the Examiner initial and return a copy of the enclosed PTO-1449.

Respectfully submitted,

PILLSBURY WINTHROP SHAW PITTMAN LLP

By /D. Benjamin Esplin/
D. Benjamin Esplin
Registration No.: 58,297

P. O. Box 10500
McLean, Virginia 22102
Telephone: (703) 770-7900
Telefax: (703) 770-7901

Customer No.
00909

Dated: July 1, 2013

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

NOTICE OF ALLOWANCE AND FEE(S) DUE

909 7590 04/18/2013

Pillsbury Winthrop Shaw Pittman, LLP (NV)
PO Box 10500
McLean, VA 22102

EXAMINER
LIN, KENNY S

ART UNIT	PAPER NUMBER
2478	

DATE MAILED: 04/18/2013

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/932,553	10/31/2007	John Albert Kembel	043311-0381668	2604

TITLE OF INVENTION: DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$890	$0	$0	$890	07/18/2013

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.

THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE DOES NOT REFLECT A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE IN THIS APPLICATION. IF AN ISSUE FEE HAS PREVIOUSLY BEEN PAID IN THIS APPLICATION (AS SHOWN ABOVE), THE RETURN OF PART B OF THIS FORM WILL BE CONSIDERED A REQUEST TO REAPPLY THE PREVIOUSLY PAID ISSUE FEE TOWARD THE ISSUE FEE NOW DUE.

HOW TO REPLY TO THIS NOTICE:

I. Review the ENTITY STATUS shown above. If the ENTITY STATUS is shown as SMALL or MICRO, verify whether entitlement to that entity status still applies.

If the ENTITY STATUS is the same as shown above, pay the TOTAL FEE(S) DUE shown above.

If the ENTITY STATUS is changed from that shown above, on PART B - FEE(S) TRANSMITTAL, complete section number 5 titled "Change in Entity Status (from status indicated above)".

For purposes of this notice, small entity fees are 1/2 the amount of undiscounted fees, and micro entity fees are 1/2 the amount of small entity fees.

II. PART B - FEE(S) TRANSMITTAL, or its equivalent, must be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed and an extra copy of the form should be submitted. If an equivalent of Part B is filed, a request to reapply a previously paid issue fee must be clearly made, and delays in processing may occur due to the difficulty in recognizing the paper as an equivalent of Part B.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Mail Stop ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Utility patents issuing on applications filed on or after Dec. 12, 1980 may require payment of maintenance fees. It is patentee's responsibility to ensure timely payment of maintenance fees when due.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), to: Mail **Mail Stop ISSUE FEE**
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450
or Fax (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

909 7590 04/18/2013
Pillsbury Winthrop Shaw Pittman, LLP (NV)
PO Box 10500
McLean, VA 22102

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission
I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being facsimile transmitted to the USPTO (571) 273-2885, on the date indicated below.

(Depositor's name)
(Signature)
(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/932,553	10/31/2007	John Albert Kembel	043311-0381668	2604

TITLE OF INVENTION: DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	SMALL	$890	$0	$0	$890	07/18/2013

EXAMINER	ART UNIT	CLASS-SUBCLASS
LIN, KENNY S	2478	709-217000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).
- ❑ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.
- ❑ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-02 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
(1) the names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.
1 _____
2 _____
3 _____

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document has been filed for recordation as set forth in 37 CFR 3.11. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE (B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ❑ Individual ❑ Corporation or other private group entity ❑ Government

4a. The following fee(s) are submitted:
- ❑ Issue Fee
- ❑ Publication Fee (No small entity discount permitted)
- ❑ Advance Order - # of Copies _____

4b. Payment of Fee(s): **(Please first reapply any previously paid issue fee shown above)**
- ❑ A check is enclosed.
- ❑ Payment by credit card. Form PTO-2038 is attached.
- ❑ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment, to Deposit Account Number _____ (enclose an extra copy of this form).

5. **Change in Entity Status** (from status indicated above)

❑ Applicant certifying micro entity status. See 37 CFR 1.29

NOTE: Absent a valid certification of Micro Entity Status (see form PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.

❑ Applicant asserting small entity status. See 37 CFR 1.27

NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.

❑ Applicant changing to regular undiscounted fee status.

NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: The Issue Fee and Publication Fee (if required) will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the United States Patent and Trademark Office.

Authorized Signature _____ Date _____

Typed or printed name _____ Registration No. _____

This collection of information is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, Virginia 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450.

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/932,553	10/31/2007	John Albert Kembel	043311-0381668	2604

909 7590 04/18/2013

Pillsbury Winthrop Shaw Pittman, LLP (NV)
PO Box 10500
McLean, VA 22102

EXAMINER	
LIN, KENNY S	

ART UNIT	PAPER NUMBER
2478	

DATE MAILED: 04/18/2013

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)
(application filed on or after May 29, 2000)

The Patent Term Adjustment to date is 870 day(s). If the issue fee is paid on the date that is three months after the mailing date of this notice and the patent issues on the Tuesday before the date that is 28 weeks (six and a half months) after the mailing date of this notice, the Patent Term Adjustment will be 870 day(s).

If a Continued Prosecution Application (CPA) was filed in the above-identified application, the filing date that determines Patent Term Adjustment is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) WEB site (http://pair.uspto.gov).

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Customer Service Center of the Office of Patent Publication at 1-(888)-786-0101 or (571)-272-4200.

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Notice of Allowability	Application No. 11/932,553	Applicant(s) KEMBEL ET AL.	
	Examiner KENNY LIN	Art Unit 2478	AIA (First Inventor to File) Status No

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☒ This communication is responsive to *4/5/2013*.
   ☐ A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on_____.
2. ☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on _____; the restriction requirement and election have been incorporated into this action.
3. ☒ The allowed claim(s) is/are *1-24*. As a result of the allowed claim(s), you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see http://www.uspto.gov/patents/init_events/pph/index.jsp or send an inquiry to PPHfeedback@uspto.gov .
4. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

   Certified copies:

   a) ☐ All b) ☐ Some *c) ☐ None of the:
   1. ☐ Certified copies of the priority documents have been received.
   2. ☐ Certified copies of the priority documents have been received in Application No. _____ .
   3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____.

   Interim copies:

   a) ☐ All b) ☐ Some c) ☐ None of the: Interim copies of the priority documents have been received.

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

5. ☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
   ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____.

   Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).
6. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1. ☐ Notice of References Cited (PTO-892)
2. ☐ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date _____
3. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material
4. ☐ Interview Summary (PTO-413), Paper No./Mail Date _____ .
5. ☐ Examiner's Amendment/Comment
6. ☐ Examiner's Statement of Reasons for Allowance
7. ☒ Other *TD approved*.

/Kenny S Lin/
Primary Examiner, Art Unit 2478

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	11932553	KEMBEL ET AL.
	Examiner	Art Unit
	BENJAMIN R BRUCKART	2446

CPC- SEARCHED		
Symbol	Date	Examiner

CPC COMBINATION SETS - SEARCHED		
Symbol	Date	Examiner

US CLASSIFICATION SEARCHED			
Class	Subclass	Date	Examiner
709	217	5/13/2010	brb
709	217	9/7/2010	brb
709	217 219	3/18/13	KL
709	217 219	4/9/13	KL

SEARCH NOTES		
Search Notes	Date	Examiner
Text search of East (USPat, USPG_Pub, JPO, EPO, Derwent, IBM_TDB) see attached history	5/13/2010	brb
Inventor search and limited class search of 709/203 and 715/502	5/13/2010	brb
Text search of East (USPat, USPG_Pub, JPO, EPO, Derwent, IBM_TDB) see attached history	9/7/2010	brb
Inventor search and limited class search of 709/203 and 715/502	9/7/2010	brb
EAST keyword search	8/29/12	KL
IDS considered	3/18/13	KL
EAST update search	3/18/13	KL
limited class search	3/18/13	KL
update search	4/9/13	KL

INTERFERENCE SEARCH

US Class/ CPC Symbol	US Subclass / CPC Group	Date	Examiner
709	217 219	4/9/13	KL

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	11932553	KEMBEL ET AL.
	Examiner	**Art Unit**
	KENNY LIN	2478

CPC		
Symbol	**Type**	**Version**

CPC Combination Sets				
Symbol	**Type**	**Set**	**Ranking**	**Version**

US ORIGINAL CLASSIFICATION		INTERNATIONAL CLASSIFICATION						
CLASS	**SUBCLASS**	**CLAIMED**					**NON-CLAIMED**	
709	217	G	0	6	F	15 / 16 (2006.0)		
CROSS REFERENCE(S)								
CLASS	**SUBCLASS (ONE SUBCLASS PER BLOCK)**							
709	219							

NONE (Assistant Examiner)	(Date)	**Total Claims Allowed:** 24	
/KENNY LIN/ Primary Examiner.Art Unit 2478 (Primary Examiner)	04/09/2013 (Date)	O.G. Print Claim(s) 1	O.G. Print Figure 6

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	11932553	KEMBEL ET AL.
	Examiner	**Art Unit**
	KENNY LIN	2478

NONE (Assistant Examiner)	(Date)	**Total Claims Allowed:** 24	
/KENNY LIN/ Primary Examiner.Art Unit 2478 (Primary Examiner)	04/09/2013 (Date)	O.G. Print Claim(s) 1	O.G. Print Figure 6

U.S. Patent and Trademark Office

Part of Paper No. 20130409

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	11932553	KEMBEL ET AL.
	Examiner	**Art Unit**
	KENNY LIN	2478

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☒ T.D. ☐ R.1.47

Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original
1	1	19	17												
3	2	20	18												
4	3	21	19												
5	4	22	20												
6	5	23	21												
7	6	24	22												
8	7	2	23												
9	8	14	24												
10	9														
11	10														
12	11														
13	12														
15	13														
16	14														
17	15														
18	16														

NONE (Assistant Examiner)	(Date)	**Total Claims Allowed:** 24	
/KENNY LIN/ Primary Examiner.Art Unit 2478 (Primary Examiner)	04/09/2013 (Date)	O.G. Print Claim(s) 1	O.G. Print Figure 6

OK TO ENTER: /KL/

Attorney Docket: 043311-0381668

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE PATENT APPLICATION OF	:	JOHN ALBERT KEMBEL, *et al.*
SERIAL NO.	:	11/932,553
FILING DATE	:	October 31, 2007
ART UNIT	:	2478
EXAMINER	:	K.S. LIN
FOR	:	DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES

RESPONSE

Mail Stop AF
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

In response to the Office Action mailed **March 25, 2013** ("Office Action"), please consider the following amendments and remarks in the above-identified application:

Amendments to the Claims are reflected in the listing of claims beginning on page **2** of this paper.

Remarks/Arguments begin on page **8** of this paper.

It is believed that no extensions of time or fees for net addition of claims are required beyond those that may otherwise be provided for in documents accompanying this paper. However, if additional extensions of time are necessary to prevent abandonment of this application, then such extensions of time are hereby petitioned for under 37 C.F.R. § 1.136(a), and any fees required therefore (including fees for net addition of claims) are hereby authorized to be charged to our Deposit Account No. 033975 (Ref. No. 043311-0381668).

Attorney Docket: 043311-0381668

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE PATENT APPLICATION OF	:	JOHN ALBERT KEMBEL, *et al.*
SERIAL NO.	:	11/932,553
FILING DATE	:	October 31, 2007
ART UNIT	:	2478
EXAMINER	:	K.S. LIN
FOR	:	DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES

RESPONSE

Mail Stop AF
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

In response to the Office Action mailed **March 25, 2013** ("Office Action"), please consider the following amendments and remarks in the above-identified application:

Amendments to the Claims are reflected in the listing of claims beginning on page **2** of this paper.

Remarks/Arguments begin on page **8** of this paper.

It is believed that no extensions of time or fees for net addition of claims are required beyond those that may otherwise be provided for in documents accompanying this paper. However, if additional extensions of time are necessary to prevent abandonment of this application, then such extensions of time are hereby petitioned for under 37 C.F.R. § 1.136(a), and any fees required therefore (including fees for net addition of claims) are hereby authorized to be charged to our Deposit Account No. 033975 (Ref. No. 043311-0381668).

CLAIM AMENDMENTS

This **listing of claims** will replace all prior versions, and listings, of claims in the application:

1. **(Currently Amended)** A client computing device configured to access content over a network, the client computing device comprising:

electronic storage configured to store networked information monitor template associated with a networked information monitor, the networked information monitor template having therein a definition of a viewer graphical user interface having a frame within which time-varying content in a web browser-readable language may be presented on a display associated with the client computing device, wherein the frame of the viewer graphical user interface lacks controls for enabling a user to specify a network location at which content for the networked information monitor is available; and

one or more processors configured to execute one or more computer program modules, the one or more computer program modules being configured to access the networked information monitor defined by the networked information monitor template, wherein accessing the networked information monitor defined by the networked information monitor template results in:

transmission, over a network to a web server at a network location, of a content request for content to be displayed within the frame of the viewer graphical user interface defined by the networked information monitor template;

reception, over the network from the web server at the network location, of content transmitted from the web server in response to the content request, the content being time-varying;

presentation, on the display, of the viewer graphical user interface defined by the networked information monitor template outside of and separate from any graphical user interface of any other application; and

presentation, on the display within the frame of the viewer graphical user interface defined by the networked information monitor, of the time-varying content received from the web server.

2. **(Currently Amended)** The client computing device of claim 23, wherein said modification corresponding to the received one or more elements comprises a modification to an image defined by the networked information monitor template as forming a part of said viewer graphical user interface.

3. **(Previously Presented)** The client computing device of claim 23, wherein the correspondence between the modification and the received one or more elements is defined by the networked information monitor template.

4. **(Previously Presented)** The client computing device of claim 23, wherein the one or more computer program modules and the networked information monitor template are configured such that modifying the feature of the viewer graphical user interface comprises adjusting a size of the frame of the viewer graphical user interface.

5. **(Previously Presented)** The client computing device of claim 23, wherein the one or more computer program modules and the networked information monitor template are configured such that modifying the feature of the viewer graphical user interface comprises changing a color of a frame border or background of the viewer graphical user interface.

6. **(Previously Presented)** The client computing device of claim 23, wherein the one or more computer program modules and the networked information monitor template are configured such that modifying the feature of the viewer graphical user interface comprises modifying text of the viewer graphical user interface in a manner defined by the networked information monitor template.

7. **(Previously Presented)** The client computing device of claim 1, wherein the networked information monitor template includes a markup language file.

8. **(Previously Presented)** The client computing device of claim 1, wherein one

or more computer program modules are configured such that the time-varying content is received from the web server over the network according to the TCP/IP protocol.

9. **(Previously Presented)** The client computing device of claim 1, wherein the network location corresponds to a uniform resource locator included in the networked information monitor template.

10. **(Previously Presented)** The client computing device of claim 9, wherein the one or more computer program modules are further configured such that accessing the networked information monitor defined by the networked information monitor template results in transmission of the content request to the uniform resource locator included in the networked information monitor template, and the content request being transmitted according to the TCP/IP protocol over the network.

11. **(Currently Amended)** The client computing device of claim 1, wherein the one or more computer program modules are further configured:

to transmit, over the network to [[an]] a networked information monitor server, a request for the networked information monitor template;

to receive, from the networked information monitor server over the network, the networked information monitor template; and

to store the networked information monitor template to the electronic storage.

12. **(Currently Amended)** A computer-implemented method of accessing ~~access~~ content over a network on a client computing device, the client computing device having electronic storage and one or more processors configured to execute one or more computer program modules, the client method comprising:

storing, to the electronic storage, a networked information monitor template associated with a networked information monitor, the networked information monitor template having therein a definition of a viewer graphical user interface having a frame within which time-varying content in a web browser-readable language may be presented on a display associated with the client computing device, wherein the frame

of the viewer graphical user interface lacks controls for enabling a user to specify a network location at which content for the networked information monitor is available;

accessing the networked information monitor defined by the networked information monitor template, wherein accessing the networked information monitor defined by the networked information monitor template results in:

transmission, over a network to a web server at a network location, of a content request for content to be displayed in the viewer graphical user interface defined by the networked information monitor template;

reception, over the network from the web server at the network location, of content transmitted from the web server in response to the content request, the content being time-varying;

presentation, on the display, of the viewer graphical user interface defined by the application media package template outside of and separate from any graphical user interface of any other application; and

presentation, on the display within the frame of the viewer graphical user interface defined by the networked information monitor, of the time-varying content received from the web server.

13. **(Previously Presented)** The method of claim 24, wherein said modification corresponding to the received one or more elements comprises a modification to an image defined by the networked information monitor template as forming a part of said viewer graphical user interface.

14. **(Previously Presented)** The method of claim 24, wherein the correspondence between the modification and the received one or more elements is defined by the networked information monitor template.

15. **(Previously Presented)** The method of claim 24, wherein modifying the feature of the viewer graphical user interface comprises adjusting a size of the frame of the viewer graphical user interface.

16. **(Previously Presented)** The method of claim 24, wherein modifying the feature of the viewer graphical user interface comprises changing a color of a frame border or background of the viewer graphical user interface.

17. **(Previously Presented)** The method of claim 24, wherein modifying the feature of the viewer graphical user interface comprises modifying text of the viewer graphical user interface in a manner defined by the networked information monitor template.

18. **(Previously Presented)** The method of claim 12, wherein the networked information monitor template includes a markup language file, and wherein storing the networked information monitor template comprises storing the markup language file.

19. **(Previously Presented)** The method of claim 12, wherein the time-varying content is received from the web server over the network according to the TCP/IP protocol.

20. **(Previously Presented)** The method of claim 12, wherein the network location corresponds to a uniform resource locator included in the networked information monitor template.

21. **(Previously Presented)** The method of claim 20, wherein accessing the networked information monitor defined by the networked information monitor template results in transmission of the content request to the uniform resource locator included in the networked information monitor template, and the content request being transmitted according to the TCP/IP protocol over the network.

22. **(Previously Presented)** The method of claim 12, further comprising:

prior to storing the networked information monitor template to the electronic storage, transmitting, over the network to a networked information monitor server, a request for the networked information monitor template; and

receiving, from the networked information monitor server over the network, the networked information monitor template.

23. **(Previously Presented)** The method of claim 1, further comprising, responsive to reception of one or more elements included in the received time-varying content, modifying a feature of said viewer graphical user interface defined by the networked information monitor template in accordance with a modification corresponding to the received one or more elements.

24. **(Previously Presented)** The method of claim 12, responsive to reception of one or more elements included in the received time-varying content, modifying a feature of said viewer graphical user interface defined by the networked information monitor template in accordance with a modification corresponding to the received one or more elements.

REMARKS

By this paper, proposed amendments for claims 1-2 and 11-12 are presented above, and no claims are added or cancelled. Accordingly, upon entry of this amendment, claims 1-24 will be all of the pending claims. The amendments proposed above address formal aspects of the claims, and do not materially impact the scope of any claim. As such, these amendments should be entered under 37 C.F.R. § 1.116. In view of the foregoing amendments and the following remarks, reconsideration and allowance of all the pending claims is anticipated.

Double Patenting

Claims 1-24 currently stand provisionally rejected on the ground of nonstatutory obviousness-type double patenting as allegedly being unpatentable over the claims in each of U.S. Patent Application Nos. 11/932,286 and 11/932,427. Without acknowledging the propriety of these rejections, and solely in an effort to expedite prosecution, Applicants have filed a Terminal Disclaimer obviating these rejections.

CONCLUSION

Having addressed each of the foregoing objections and rejections, it is respectfully submitted that a full and complete response has been made to the outstanding Office Action and, as such, the application is in condition for allowance. Notice to that effect is anticipated.

If the Examiner believes, for any reason, that personal communication will expedite prosecution of this application, the Examiner is invited to telephone the undersigned at the number provided.

Date: April 5, 2013

Respectfully submitted,

By: /D. Benjamin Esplin/
D. Benjamin Esplin
Reg. No. 58,297

Customer No. 00909

PILLSBURY WINTHROP SHAW PITTMAN LLP
1650 Tysons Boulevard
McLean, Virginia 22102
619-234-5000

Electronic Acknowledgement Receipt

EFS ID:	15449731
Application Number:	11932553
International Application Number:	
Confirmation Number:	2604
Title of Invention:	Displaying Time-Varying Internet Based Data Using Application Media Packages
First Named Inventor/Applicant Name:	John Albert Kembel
Customer Number:	909
Filer:	D. Benjamin Esplin/Maxine Cutietta
Filer Authorized By:	D. Benjamin Esplin
Attorney Docket Number:	043311-0381668
Receipt Date:	05-APR-2013
Filing Date:	31-OCT-2007
Time Stamp:	17:48:17
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		0381668_Response_Filed04052013.pdf	49198 86ad6e8421c2077399b6f0f072ac0026d1d23f06	yes	9

	Multipart Description/PDF files in .zip description		
	Document Description	**Start**	**End**
	Amendment After Final	1	1
	Claims	2	7
	Applicant Arguments/Remarks Made in an Amendment	8	9
Warnings:			
Information:			
	Total Files Size (in bytes):	49198	

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Doc Code: DIST.E.FILE **Document Description: Electronic Terminal Disclaimer - Filed**	PTO/SB/25 U.S. Patent and Trademark Office Department of Commerce

Electronic Petition Request	**TERMINAL DISCLAIMER TO OBVIATE A PROVISIONAL DOUBLE PATENTING REJECTION OVER A PENDING "REFERENCE" APPLICATION**
Application Number	11932553
Filing Date	31-Oct-2007
First Named Inventor	John Kembel
Attorney Docket Number	043311-0381668
Title of Invention	Displaying Time-Varying Internet Based Data Using Application Media Packages

☒ Filing of terminal disclaimer does not obviate requirement for response under 37 CFR 1.111 to outstanding Office Action

☒ This electronic Terminal Disclaimer is not being used for a Joint Research Agreement.

Owner	Percent Interest
MAINSTREAM SCIENTIFIC, LLC	100%

The owner(s) of percent interest listed above in the instant application hereby disclaims, except as provided below, the terminal part of the statutory term of any patent granted on the instant application which would extend beyond the expiration date of the full statutory term of any patent granted on pending reference Application Number(s)

11932286 filed on 10/31/2007

11932427 filed on 10/31/2007

as the term of any patent granted on said reference application may be shortened by any terminal disclaimer filed prior to the grant of any patent on the pending reference application. The owner hereby agrees that any patent so granted on the instant application shall be enforceable only for and during such period that it and any patent granted on the reference application are commonly owned. This agreement runs with any patent granted on the instant application and is binding upon the grantee, its successors or assigns.

In making the above disclaimer, the owner does not disclaim the terminal part of any patent granted on the instant application that would extend to the expiration date of the full statutory term of any patent granted on said reference application, "as the term of any patent granted on said reference application may be shortened by any terminal disclaimer filed prior to the grant of any patent on the pending reference application," in the event that any such patent granted on the pending reference application: expires for failure to pay a maintenance fee, is held unenforceable, is found invalid by a court of competent jurisdiction, is statutorily disclaimed in whole or terminally disclaimed under 37 CFR 1.321, has all claims canceled by a reexamination certificate, is reissued, or is in any manner terminated prior to the expiration of its full statutory term as shortened by any terminal disclaimer filed prior to its grant.

◉ Terminal disclaimer fee under 37 CFR 1.20(d) is included with Electronic Terminal Disclaimer request.

○	I certify, in accordance with 37 CFR 1.4(d)(4), that the terminal disclaimer fee under 37 CFR 1.20(d) required for this terminal disclaimer has already been paid in the above-identified application.

○ Applicant claims SMALL ENTITY status. See 37 CFR 1.27.

○ Applicant is no longer claiming SMALL ENTITY status. See 37 CFR 1.27(g)(2).

◉ Applicant(s) status remains as SMALL ENTITY.

○ Applicant(s) status remains as other than SMALL ENTITY.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

THIS PORTION MUST BE COMPLETED BY THE SIGNATORY OR SIGNATORIES

I certify, in accordance with 37 CFR 1.4(d)(4) that I am:

◉ An attorney or agent registered to practice before the Patent and Trademark Office who is of record in this application

Registration Number 58297

○ A sole inventor

○ A joint inventor; I certify that I am authorized to sign this submission on behalf of all of the inventors

○ A joint inventor; all of whom are signing this request

○ The assignee of record of the entire interest that has properly made itself of record pursuant to 37 CFR 3.71

Signature	/D. Benjamin Esplin/
Name	D. Benjamin Esplin

*Statement under 37 CFR 3.73(b) is required if terminal disclaimer is signed by the assignee (owner).
Form PTO/SB/96 may be used for making this certification. See MPEP § 324.

Electronic Patent Application Fee Transmittal

Application Number:	11932553
Filing Date:	31-Oct-2007
Title of Invention:	Displaying Time-Varying Internet Based Data Using Application Media Packages
First Named Inventor/Applicant Name:	John Albert Kembel
Filer:	D. Benjamin Esplin/Maxine Cutietta
Attorney Docket Number:	043311-0381668

Filed as Small Entity

Utility under 35 USC 111(a) Filing Fees

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Statutory or Terminal Disclaimer	1814	1	160	160
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Miscellaneous:				
			Total in USD ($)	**160**

Doc Code: DISQ.E.FILE
Document Description: Electronic Terminal Disclaimer – Approved

Application No.: 11932553

Filing Date: 31-Oct-2007

Applicant/Patent under Reexamination: Kembel et al.

Electronic Terminal Disclaimer filed on April 5, 2013

☒ APPROVED

This patent is subject to a terminal disclaimer

☐ DISAPPROVED

Approved/Disapproved by: Electronic Terminal Disclaimer automatically approved by EFS-Web

U.S. Patent and Trademark Office

Electronic Acknowledgement Receipt	
EFS ID:	15449977
Application Number:	11932553
International Application Number:	
Confirmation Number:	2604
Title of Invention:	Displaying Time-Varying Internet Based Data Using Application Media Packages
First Named Inventor/Applicant Name:	John Albert Kembel
Customer Number:	909
Filer:	D. Benjamin Esplin/Maxine Cutietta
Filer Authorized By:	D. Benjamin Esplin
Attorney Docket Number:	043311-0381668
Receipt Date:	05-APR-2013
Filing Date:	31-OCT-2007
Time Stamp:	18:14:09
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	Deposit Account
Payment was successfully received in RAM	$160
RAM confirmation Number	5456
Deposit Account	033975
Authorized User	
The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows: Charge any Additional Fees required under 37 C.F.R. Section 1.16 (National application filing, search, and examination fees) Charge any Additional Fees required under 37 C.F.R. Section 1.17 (Patent application and reexamination processing fees)	

Charge any Additional Fees required under 37 C.F.R. Section 1.19 (Document supply fees)

Charge any Additional Fees required under 37 C.F.R. Section 1.20 (Post Issuance fees)

Charge any Additional Fees required under 37 C.F.R. Section 1.21 (Miscellaneous fees and charges)

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Electronic Terminal Disclaimer-Filed	eTerminal-Disclaimer.pdf	36811 ac7cefad8ea928a08fc38bd0bde8a833360a6cc0	no	2
Warnings:					
Information:					
2	Fee Worksheet (SB06)	fee-info.pdf	30441 0aee7af66f68ac99cdd96c45112fc6347fd9df37	no	2
Warnings:					
Information:					
		Total Files Size (in bytes):	67252		

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

<u>New Applications Under 35 U.S.C. 111</u>
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

<u>National Stage of an International Application under 35 U.S.C. 371</u>
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

<u>New International Application Filed with the USPTO as a Receiving Office</u>
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Electronic Patent Application Fee Transmittal	
Application Number:	11932553
Filing Date:	31-Oct-2007
Title of Invention:	Displaying Time-Varying Internet Based Data Using Application Media Packages
First Named Inventor/Applicant Name:	John Albert Kembel
Filer:	D. Benjamin Esplin/Maxine Cutietta
Attorney Docket Number:	043311-0381668
Filed as Small Entity	

Utility under 35 USC 111(a) Filing Fees

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Miscellaneous:				
Statutory or Terminal Disclaimer	1814	1	160	160
		Total in USD ($)		**160**

Electronic Acknowledgement Receipt	
EFS ID:	15450064
Application Number:	11932553
International Application Number:	
Confirmation Number:	2604
Title of Invention:	Displaying Time-Varying Internet Based Data Using Application Media Packages
First Named Inventor/Applicant Name:	John Albert Kembel
Customer Number:	909
Filer:	D. Benjamin Esplin/Maxine Cutietta
Filer Authorized By:	D. Benjamin Esplin
Attorney Docket Number:	043311-0381668
Receipt Date:	05-APR-2013
Filing Date:	31-OCT-2007
Time Stamp:	18:16:27
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	Deposit Account
Payment was successfully received in RAM	$160
RAM confirmation Number	5491
Deposit Account	033975
Authorized User	
The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows: Charge any Additional Fees required under 37 C.F.R. Section 1.16 (National application filing, search, and examination fees) Charge any Additional Fees required under 37 C.F.R. Section 1.17 (Patent application and reexamination processing fees)	

Charge any Additional Fees required under 37 C.F.R. Section 1.19 (Document supply fees) Charge any Additional Fees required under 37 C.F.R. Section 1.20 (Post Issuance fees) Charge any Additional Fees required under 37 C.F.R. Section 1.21 (Miscellaneous fees and charges)

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Fee Worksheet (SB06)	fee-info.pdf	30439 ac56bdebd221887fec0bad588117eeb7a8043462	no	2

Warnings:	
Information:	
Total Files Size (in bytes):	30439

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PTO/SB/06 (09-11)
Approved for use through 1/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD Substitute for Form PTO-875	Application or Docket Number 11/932,553	Filing Date 10/31/2007	☐ To be Mailed

ENTITY: ☐ LARGE ☒ SMALL ☐ MICRO

APPLICATION AS FILED – PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA	RATE ($)	FEE ($)
☐ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A	
☐ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A	
☐ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*	X $ =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*	X $ =	
☐ APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).			
☐ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))				
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL	

APPLICATION AS AMENDED – PART II

AMENDMENT 04/05/2013

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)
Total (37 CFR 1.16(i))	* 24	Minus	** 24	= 0	X $40 =	0
Independent (37 CFR 1.16(h))	* 2	Minus	***3	= 0	X $210 =	0
☐ Application Size Fee (37 CFR 1.16(s))						
☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))						
					TOTAL ADD'L FEE	0

AMENDMENT

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)
Total (37 CFR 1.16(i))	*	Minus	**	=	X $ =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	X $ =	
☐ Application Size Fee (37 CFR 1.16(s))						
☐ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))						
					TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

LIE
/JAMES MASON/

This collection of information is required by 37 CFR 1.16. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**
If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/932,553	10/31/2007	INV001John Albert Kembel	043311-0381668	2604

909 7590 03/25/2013

Pillsbury Winthrop Shaw Pittman, LLP (NV)
PO Box 10500
McLean, VA 22102

EXAMINER	
LIN, KENNY S	
ART UNIT	PAPER NUMBER
2478	

NOTIFICATION DATE	DELIVERY MODE
03/25/2013	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

docket_ip@pillsburylaw.com
margaret.drosos@pillsburylaw.com

Office Action Summary	Application No.	Applicant(s)	
	11/932,553	KEMBEL ET AL.	
	Examiner	Art Unit	
	KENNY LIN	2478	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *05 March 2013*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on _____; the restriction requirement and election have been incorporated into this action.

4)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

5)☒ Claim(s) *1-22* is/are pending in the application.

5a) Of the above claim(s) _____ is/are withdrawn from consideration.

6)☐ Claim(s) _____ is/are allowed.

7)☒ Claim(s) *1-22* is/are rejected.

8)☐ Claim(s) _____ is/are objected to.

9)☐ Claim(s) _____ are subject to restriction and/or election requirement.

* If any claims have been determined allowable, you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see http://www.uspto.gov/patents/init_events/pph/index.jsp or send an inquiry to PPHfeedback@uspto.gov.

Application Papers

10)☐ The specification is objected to by the Examiner.

11)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *3/4/13*.

3) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .

4) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-24 are presented for examination.

Information Disclosure Statement

2. The IDS filed on 3/4/2013 is considered.

Double Patenting

3. The nonstatutory double patenting rejection is based on a judicially created doctrine grounded in public policy (a policy reflected in the statute) so as to prevent the unjustified or improper timewise extension of the "right to exclude" granted by a patent and to prevent possible harassment by multiple assignees. A nonstatutory obviousness-type double patenting rejection is appropriate where the conflicting claims are not identical, but at least one examined application claim is not patentably distinct from the reference claim(s) because the examined application claim is either anticipated by, or would have been obvious over, the reference claim(s). See, e.g., *In re Berg*, 140 F.3d 1428, 46 USPQ2d 1226 (Fed. Cir. 1998); *In re Goodman*, 11 F.3d 1046, 29 USPQ2d 2010 (Fed. Cir. 1993); *In re Longi*, 759 F.2d 887, 225 USPQ 645 (Fed. Cir. 1985); *In re Van Ornum*, 686 F.2d 937, 214 USPQ 761 (CCPA 1982); *In re Vogel*, 422 F.2d 438, 164 USPQ 619 (CCPA 1970); and *In re Thorington*, 418 F.2d 528, 163 USPQ 644 (CCPA 1969).

A timely filed terminal disclaimer in compliance with 37 CFR 1.321(c) or 1.321(d) may be used to overcome an actual or provisional rejection based on a nonstatutory double patenting ground provided the conflicting application or patent either is shown to be commonly owned

with this application, or claims an invention made as a result of activities undertaken within the scope of a joint research agreement.

Effective January 1, 1994, a registered attorney or agent of record may sign a terminal disclaimer. A terminal disclaimer signed by the assignee must fully comply with 37 CFR 3.73(b).

1. Claims 1-24 are provisionally rejected on the ground of nonstatutory obviousness-type double patenting as being unpatentable over claims 1-32 of U.S. Patent Application 11/932,286 which is currently pending.

This is a provisional obviousness-type double patenting rejection because the conflicting claims have not in fact been patented.

2. Claims 1-24 are provisionally rejected on the ground of nonstatutory obviousness-type double patenting as being unpatentable over claims 1-25 of U.S. Patent Application 11/932,427 which is currently pending.

This is a provisional obviousness-type double patenting rejection because the conflicting claims have not in fact been patented.

Response to Arguments

4. Applicant's arguments and amendment, filed on 3/5/2013, with respect to s 1-24 have been fully considered and are persuasive. The 103 rejection of claims 1-22 has been withdrawn.

5. Applicant's arguments filed 3/5/2013 regarding the double patenting have been fully considered but they are not persuasive. The pending claims remains to contain double patenting issues with applications 11/932,286 and 11/932,427.

Conclusion

6. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Kenny Lin whose telephone number is (571) 272-3968.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Derrick Ferris can be reached on (571) 272-3123. The fax phone number for the organization where this application or proceeding is assigned is (571) 273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

/Kenny S Lin/
Primary Examiner, Art Unit 2478
March 20, 2013

Notice of References Cited	Application/Control No. 11/932,553	Applicant(s)/Patent Under Reexamination KEMBEL ET AL.	
	Examiner KENNY LIN	Art Unit 2478	Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	Classification
*	A	US-6,101,510	08-2000	Stone et al.	715/234
*	B	US-6,128,655	10-2000	Fields et al.	709/219
*	C	US-6,356,905	03-2002	Gershman et al.	705/26.8
	D	US-			
	E	US-			
	F	US-			
	G	US-			
	H	US-			
	I	US-			
	J	US-			
	K	US-			
	L	US-			
	M	US-			

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

INFORMATION DISCLOSURE CITATION IN AN APPLICATION	Docket Number	Application Number
	043311-0381668	11/932,553
	Applicant(s)	
	John Albert KEMBEL *et al.*	
	Filing Date	Group Art Unit
Form PTO-1449	October 31, 2007	2478

U.S. PATENT DOCUMENTS

*EXAMINER INITIAL	REF	DOCUMENT NO.	DATE	NAME	CLASS	SUBCLASS	FILING DATE IF APPROPRIATE
	~~1A~~	~~08/159,917~~	~~09/14/1999~~	~~Franco et al.~~			
	1B	5,599,756	02/04/1997	Matsuo (Hermann)	501	127	
	1C	5,682,511	10/28/1997	Sposato et al.	715	716	
	1D	5,745,718	04/28/1998	Cline et al.	715	777	
	1E	5,774,670	06/30/1998	Montulli	395	200.57	
	1F	5,802,530	09/01/1998	Van Hoff	715	513	
	1G	5,805,829	09/08/1998	Cohen *et al.*	395	200.32	
	1H	5,818,446	10/06/1998	Bertram et al.	715	746	
	1I	5,835,088	11/10/1998	Jaaskelainen, Jr.	715	803	
	1J	5,860,068	01/12/1999	Cook	705	26.81	
	1K	5,864,676	01/26/1999	Beer et al.	709	229	
	1L	5,864,868	01/26/1999	Contois	707	104	
	1M	5,896,533	04/20/1999	Ramos et al.	395	680	
	1N	5,918,237	06/29/1999	Montalbano	715	206	
	1O	5,919,247	07/06/1999	Van Hoff et al.	709	217	
	1P	5,922,044	07/13/1999	Banthia	709	203	
	1Q	5,923,845	07/13/1999	Kamiya et al.	709	206	
	1R	5,923,885	07/13/1999	Johnson et al.	717	176	
	1S	5,959,621	09/28/1999	Nawaz et al.	345	329	
	1T	5,966,715	10/12/1999	Sweeney et al.	707	203	
	1U	5,973,692	10/26/1999	Knowlton et al.	345	348	
	1V	5,974,546	10/26/1999	Anderson	713	2	
	1W	6,034,652	03/07/2000	Freiberger et al.	715	730	

FOREIGN PATENT DOCUMENTS

	REF	DOCUMENT NO.	DATE	COUNTRY	CLASS	SUBCLASS	Translations YES	Translations NO

OTHER DOCUMENTS *(Including Author, Title, Date, Pertinent Pages, Etc.)*

	1X	Patent Application entitled "Parallel Web Sites", Serial No. 09/192,633, Filed November 16, 1998.
	1Y	Microsoft Computer Dictionary, Fifth Edition, 2002, Definition of "Web Browser".
	1Z	WILLIAMS, Margot, "Cyberspace Calendars: The Web's Growing Date Base", November 30, 1998, The Washington Post, page 1.
	1AA	BOTT, Ed., *et al.*, "Special Edition Using Windows 95 with Internet Explorer 4.0", Publisher: Que, February 17, 1998, pages 585 and 435.

EXAMINER	DATE CONSIDERED
/Kenny Lin/	03/18/2013

EXAMINER: Initial if citation considered, whether or not citation is in conformance with MPEP Section 609; Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to Applicant.

403833160v1

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /KL/

INFORMATION DISCLOSURE CITATION IN AN APPLICATION	Docket Number	Application Number
	043311-0381668	11/932,553
	Applicant(s)	
	John Albert KEMBEL *et al.*	
	Filing Date	Group Art Unit
Form PTO-1449	October 31, 2007	2478

U.S. PATENT DOCUMENTS

*EXAMINER INITIAL	REF	DOCUMENT NO.	DATE	NAME	CLASS	SUBCLASS	FILING DATE IF APPROPRIATE
	2A	6,061,695	05/09/2000	Slivka *et al.*	707	513	
	2B	6,088,717	07/11/2000	Reed et al.	709	201	
	2C	6,091,411	07/18/2000	Straub et al.	715	747	
	2D	6,091,412	07/18/2000	Simonoff *et al.*	345	335	
	2E	6,104,391	08/15/2000	Johnston, Jr. et al.	715	745	
	2F	6,105,063	08/15/2000	Hayes, Jr.	709	223	
	2G	6,115,040	09/05/2000	Bladow et al.	345	335	
	2H	6,192,407	02/20/2001	Smith et al.	709	229	
	2I	6,215,490	04/10/2001	Kaply	715	788	
	2J	6,240,555	05/29/2001	Shoff *et al.*	725	110	
	2K	6,278,449	08/21/2001	Sugiarto et al.	715	826	
	2L	6,286,034	09/04/2001	Sato et al.	709	204	
	2M	6,289,362	09/11/2001	Van Der Meer	715	273	
	2N	6,292,186	09/18/2001	Lehman et al.	345	335	
	2O	6,317,759	11/13/2001	Osmond	715	513	
	2P	6,339,826	01/15/2002	Hayes, Jr. et al.	713	166	
	2Q	6,342,907	01/29/2002	Petty et al.	345	762	
	2R	6,370,552	04/09/2002	Bloomfield	715	513	
	2S	6,374,273	04/16/2002	Webster	707	513	
	2T	6,385,596	05/07/2002	Wiser et al.	705	51	
	2U	6,401,134	06/04/2002	Razavi *et al.*	709	310	
	2V	6,414,677	07/02/2002	Robertson et al.	345	419	
	2W	6,434,598	08/13/2002	Gish	709	203	

FOREIGN PATENT DOCUMENTS

	REF	DOCUMENT NO.	DATE	COUNTRY	CLASS	SUBCLASS	Translations YES	Translations NO

OTHER DOCUMENTS *(Including Author, Title, Date, Pertinent Pages, Etc.)*

	2X	McFEDRIES, Paul, "Windows 98 Unleashed", Publisher: Sams, May 12, 1998, pages 594-596.
	2Y	McFEDRIES, Paul, "Windows 98 Unleashed", Publisher: Sams Publishing, May 12, 1998, pages xix, xx, xxi, xxiv, 2-4, 44-46, 69, 70, 79, 80, 97-116 ,148, 158-163, 251, 551, 787-792, 799-807, 885, 899, 900, and 904-906.
	2Z	McCRICKARD, D. Scott, *et al.*, "Supporting Information Awareness Using Animated Widgets", Proceedings of the 7th USENIX Tel/Tk Conference, February 14-18, 2000, 12 pages.

EXAMINER	DATE CONSIDERED
/Kenny Lin/	03/18/2013

EXAMINER: Initial if citation considered, whether or not citation is in conformance with MPEP Section 609; Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to Applicant.

403833160v1

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /KL/

INFORMATION DISCLOSURE CITATION IN AN APPLICATION	Docket Number	Application Number
	043311-0381668	11/932,553
	Applicant(s)	
	John Albert KEMBEL *et al.*	
	Filing Date	Group Art Unit
Form PTO-1449	October 31, 2007	2478

U.S. PATENT DOCUMENTS

*EXAMINER INITIAL	REF	DOCUMENT NO.	DATE	NAME	CLASS	SUBCLASS	FILING DATE IF APPROPRIATE
	3A	6,452,609	09/17/2002	Katinsky et al.	345	716	
	3B	6,453,348	09/17/2002	Barnier *et al.*	709	225	
	3C	6,476,833	11/05/2002	Moshfeghi	345	854	
	3D	6,487,663	11/26/2002	Jaisimha *et al.*	713	193	
	3E	6,496,203	12/17/2002	Beaumont et al.	345	762	
	3F	6,510,466	01/21/2003	Cox *et al.*	709	229	
	3G	6,606,657	08/12/2003	Zilberstein et al.	709	224	
	3H	6,662,341	12/09/2003	Cooper et al.	715	513	
	3I	6,681,368	01/20/2004	Kawabata	715	501.1	
	3J	6,687,745	02/03/2004	Franco et al.	709	219	
	3K	6,691,130	02/10/2004	Kawasaki et al.	707	102	
	3L	6,694,484	02/17/2004	Mueller	715	513	
	3M	6,718,015	04/06/2004	Berstis	379	88.17	
	3N	6,724,403	04/20/2004	Santoro et al.	345	765	
	3O	6,757,716	06/29/2004	Blegen et al.	709	217	
	3P	6,766,454	07/20/2004	Riggins	713	185	
	3Q	6,819,343	11/16/2004	Sobeski et al.	715	848	
	3R	6,834,302	12/21/2004	Harvell	709	224	
	3S	6,879,994	04/12/2005	Matsliach et al.	709	204	
	3T	7,039,857	05/02/2006	Beck *et al.*	715	500.1	
	3U	7,039,859	05/02/2006	Sundaresan	715	513	
	3V	7,076,737	07/11/2006	Abbott et al.	715	744	
	3W	7,222,303	05/22/2007	Oren et al.	715	744	

FOREIGN PATENT DOCUMENTS

	REF	DOCUMENT NO.	DATE	COUNTRY	CLASS	SUBCLASS	Translations YES	Translations NO

OTHER DOCUMENTS *(Including Author, Title, Date, Pertinent Pages, Etc.)*

	3X	"Streaming Internet Technologies", PR Newswire, May 18, 1999, 2 pages.
	3Y	"NewsEdge Delivers the Power of Real-Time News in a Browser", Business Wire, November 9, 1998, 3 pages.
	3Z	McCARTNEY, Terrance Paul, "End-User Construction and Configuration of Distributed Multimedia Applications", ProQuest Dissertations and Theses, 1996, 197 pages.

EXAMINER	DATE CONSIDERED
/Kenny Lin/	03/18/2013

EXAMINER: Initial if citation considered, whether or not citation is in conformance with MPEP Section 609; Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to Applicant.

403833160v1

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /KL/

INFORMATION DISCLOSURE CITATION IN AN APPLICATION	Docket Number	Application Number
	043311-0381668	11/932,553
	Applicant(s)	
	John Albert KEMBEL *et al.*	
	Filing Date	Group Art Unit
Form PTO-1449	October 31, 2007	2478

U.S. PATENT DOCUMENTS

*EXAMINER INITIAL	REF	DOCUMENT NO.	DATE	NAME	CLASS	SUBCLASS	FILING DATE IF APPROPRIATE
	4A	7,574,649	08/11/2009	Safars *et al.*	715	200	
	4B	7,756,967	07/13/2010	Kembel et al.	709	224	
	4C	7,660,868	02/09/2010	Kembel et al.	709	217	
	4D	7,792,947	09/07/2010	Kembel et al.	709	224	
	4E	8,020,083	09/13/2001	Kembel et al.	715	201	
	4F	8,346,887	01/01/2013	Kembel et al.	709	217	
	4G	2001/0042107	11/15/2001	Palm	709	218	
	4H	2002/0065896	05/30/2002	Burakoff et al.	709	206	
	4I	2002/0078136	06/20/2002	Brodsky et al.	709	203	
	4J	2002/0089526	07/11/2002	Buxton et al.	345	700	
	4K	2002/0089536	07/11/2002	Dang	345	749	
	4L	2002/0091697	07/11/2002	Huang et al.	707	10	
	4M	2002/0130900	09/19/2002	Davis	345	744	
	4N	2002/0161879	10/31/2002	Richard	709	223	
	4O	2003/0069944	04/10/2003	Barlock et al.	709	220	
	4P	2004/0041836	03/01/2004	Zaner et al.	345	751	
	4Q	2004/0165007	08/26/2004	Shafron	715	781	
	4R	2005/0273718	12/08/2005	Naas	715	745	
	4S	2008/0134018	06/05/2008	Kembel et al.	715	234	
	4T	2008/0163202	07/03/2008	Kembel et al.	717	178	
	4U	2008/0229217	09/18/2008	Kembel et al.	715	760	
	4V	2008/0040681	02/14/2008	Synstelien et al.	715	765	
	4W	2010/0235757	09/16/2010	Kembel et al.	715	745	
	4X	2010/0257442	10/07/2010	Kembel et al.	715	234	
	4Y	2012/0117479	05/10/2012	Kembel et al.	715	736	

FOREIGN PATENT DOCUMENTS

	REF	DOCUMENT NO.	DATE	COUNTRY	CLASS	SUBCLASS	Translations YES	Translations NO

OTHER DOCUMENTS *(Including Author, Title, Date, Pertinent Pages, Etc.)*

	4Z	"Microsoft Eyes Marimba's Castanet", by CNET News.com Staff, December 24, 1996, printed from <http://news.cnet.com/Microsoft-eyes-Marimbas-Castanet/2100-1001_3-257491.html>, 2 pages.
	4AA	WILLIAMS, Dennis, "Application Delivery on a Grand Scale", Network World Fusion, March 22, 1999, printed from <http://www.networkworld.com/reviews/0322revmarimba.html>, 4 pages.

EXAMINER	DATE CONSIDERED
/Kenny Lin/	03/18/2013

EXAMINER: Initial if citation considered, whether or not citation is in conformance with MPEP Section 609; Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to Applicant.

403833160v1

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /KL/

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	11932553	KEMBEL ET AL.
	Examiner	Art Unit
	BENJAMIN R BRUCKART	2446

CPC- SEARCHED		
Symbol	Date	Examiner

CPC COMBINATION SETS - SEARCHED		
Symbol	Date	Examiner

US CLASSIFICATION SEARCHED			
Class	Subclass	Date	Examiner
709	217	5/13/2010	brb
709	217	9/7/2010	brb
709	217 219	3/18/13	KL

SEARCH NOTES		
Search Notes	Date	Examiner
Text search of East (USPat, USPG_Pub, JPO, EPO, Derwent, IBM_TDB) see attached history	5/13/2010	brb
Inventor search and limited class search of 709/203 and 715/502	5/13/2010	brb
Text search of East (USPat, USPG_Pub, JPO, EPO, Derwent, IBM_TDB) see attached history	9/7/2010	brb
Inventor search and limited class search of 709/203 and 715/502	9/7/2010	brb
EAST keyword search	8/29/12	KL
IDS considered	3/18/13	KL
EAST update search	3/18/13	KL
limited class search	3/18/13	KL

INTERFERENCE SEARCH			
US Class/ CPC Symbol	US Subclass / CPC Group	Date	Examiner

EAST Search History

EAST Search History (Prior Art)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S46	2	("6101510").pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 14:10
S45	2	("5922044").pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 14:05
S44	3	("7756967" "7356569").pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 13:45
S43	68	("20020089536" | "20020091697" | "20020161879" | "5599756" | "5625781" | "5649186" | "5682511" | "5740549" | "5761662" | "5794230" | "5796393" | "5796952" | "5801702" | "5809248" | "5838906" | "5864676" | "5890172" | "5893091" | "5948061" | "5974446" | "5977964" | "5983227" | "5987513" | "5995756" | "6006252" | "6012090" | "6012098" | "6018344" | "6026433" | "6065044" | "6091411" | "6101510" | "6104391" | "6133916" | "6161112" | "6177936" | "6199082" | "6230173" | "6237030" | "6268856" | "6275854" | "6292185" | "6297819" | "6314451" | "6341305" | "6369840" | "6393407" | "6411992" | "6418440" | "6434563" | "6460029" | "6484149" | "6487566" | "6537324" | "6538673" | "6549612" | "6560639" | "6594682" | "6681368" | "6691130" | "6751606" | "6784900" | "6816880" | "6842779" | "7076737").PN. OR ("7660868").URPN.	US-PGPUB; USPAT; USOCR	OR	ON	2013/03/18 13:44
S42	2	"7660868".pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 13:32

S41	0	S40 and (@AD < "19990426" or @PRAD < "19990426")	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 13:26
S40	6	present$3 near6 content near8 separat$3 near6 (window display interface) and server near7 (template format) near6 content	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 13:26
S39	0	S38 and (@AD < "19990426" or @PRAD < "19990426")	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 13:19
S38	10	render$3 near6 content near8 separat$3 near6 (window display interface) and server near7 (template format) near6 content	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 13:19
S37	26	("5774670" "5805829" "5818446" "5919247" "5922044" "5959621" "6034652" "6061695" "6061696" "6105063" "6240555" "6339826" "6401134" "6453348" "6476833" "6487663" "6510466" "6687745" "6718015" "6724403" "6766454" "7039857" "7574649" "8346887" "20010042107" "20120117479").pn.	US-PGPUB; USPAT	OR	ON	2013/03/18 13:05
S36	52	("5774670" "5805829" "5818446" "5919247" "5922044" "5959621" "6034652" "6061695" "6061696" "6105063" "6240555" "6339826" "6401134" "6453348" "6476833" "6487663" "6510466" "6687745" "6718015" "6724403" "6766454" "7039857" "7574649" "8346887" "20010042107" "20120117479").pn.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 13:04
S35	6	60/153917	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 13:00
S34	3	S32 and (@AD < "19990426" or @PRAD < "19990426")	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 12:15
S33	0	S31 and (@AD < "19990426" or @PRAD	US-PGPUB;	OR	ON	2013/03/18

		< "19990426")	USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB			12:15
S32	19	S29 and content near4 template and retriev$3 near6 content near4 web	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 12:15
S31	15	S30 and content near4 template and retriev$3 near6 content near4 web	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 12:15
S30	3838	709/246.ccls.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 12:15
S29	11132	709/219.ccls.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 12:14
S28	0	S26 and (@AD < "19990426" or @PRAD < "19990426")	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 12:11
S27	3	S25 and (@AD < "19990426" or @PRAD < "19990426")	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 12:11
S26	14	server near4 portal and retriev$3 near6 web near6 content and content near3 template	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 12:11
S25	50	S24 and content near4 template and retriev$3 near6 content near4 web	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT;	OR	ON	2013/03/18 12:11

			IBM_TDB			
S24	12907	709/217.ccls.	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 12:11
S23	0	S22 and (@AD < "19990426" or @PRAD < "19990426")	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 12:10
S22	20	server near4 portal and retriev$3 near6 (URL or URI) near6 content and content near3 template	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 12:10
S21	2	server near4 portal and retriev$3 near6 (URL or URI) near6 content and customiz$3 near6 content with template	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 11:53
S20	2	server with retriev$3 near6 (URL or URI) near6 content and customiz$3 near6 content with template	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 11:22
S19	0	server with pars$3 near6 (URL or URI) near6 content and customiz$3 near6 content with template	US-PGPUB; USPAT; USOCR; FPRS; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2013/03/18 11:21

3/19/2013 11:20:32 AM
C:\Users\klin1\Documents\EAST\Workspaces\11932392.wsp

SUBSTITUTE SPECIFICATION (CLEAN)

DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] The present application is a continuation of and incorporates by reference U.S. Non-Provisional patent application Ser. No. 09/558,925, filed Apr. 26, 2000, now U.S. Patent No. 7,660,868, which claims priority from and incorporates by reference U.S. Provisional Application ser. nos. 60/131,083, filed Apr. 26, 1999, 60/131,114, filed Apr. 26, 1999, 60/131,115, filed Apr. 26, 1999, 60/176,687, filed Jan. 18, 2000, and 60/176,699, filed Jan. 18, 2000. The present application claims priority to U.S. Non-Provisional patent application ser. no. 09/558,925, filed Apr. 26, 2000, now U.S. Patent No. 7,660,868, and each of the aforementioned applications to which it claims priority.

[0002] The present application is also related to and incorporates by reference the following U.S. patent applications: Non-Provisional Application ser. no. 09/558,922, filed April 26, 2000, now U.S. Patent No. 7,756,967; Non-Provisional Application ser. no. 09/558,923, filed April 26, 2000; Non-Provisional Application ser. no. 09/558,924, filed April 26, 2000, now U.S. Patent No. 7,356,569; Non-Provisional Application Ser. No. xx/xxx,xxx, filed October 31, 2007 titled "Component For Accessing And Displaying Internet Content" (attorney docket no. IMS 07-05-01); Non-Provisional Application ser. no. 11/932,286, filed Oct. 31, 2007, titled "Server Including Components For Accessing And Displaying Internet Content And For Providing Same To A Client"; Non-Provisional Application ser. no.

11/932,392, filed Oct. 31, 2007, titled "Method For Accessing And Displaying Internet Content"; Non-Provisional application Ser. No. 11/932,427, filed Oct. 31, 2007, titled "Component For Coordinating The Accessing And Rendering Of An Application Media Package"; Non-Provisional Application ser. no. 11/932,456, filed Oct. 31, 2007, titled "Tracking and Tracing User Activity with Application Media Packages"; Non-Provisional Application ser. no. 11/932,585, filed Oct. 31, 2007, titled "System and Methods for Creating and Authoring Internet Content using Application Media Packages"; Non-Provisional Application ser. no. 11/932,630, filed Oct. 31, 2007, titled "Methods of Obtaining Application Media Packages"; Non-Provisional Application ser. no. 11/932,663, filed Oct. 31, 2007, titled "Indexing, Sorting, and Categorizing Dots"; Non-Provisional Application ser. no. 11/932,692, filed Oct. 31, 2007, titled "System and Methods of Messaging between Application Media Packages; and, Non-Provisional Application ser. no. 11/932,763, filed Oct. 31, 2007, titled "Component For Accessing And Displaying Internet Content In Association With a Web Browser Application".

BACKGROUND OF THE INVENTION

(03) A user operating a client computer typically accesses the Internet by using a viewer application, such as a browser to view web content provided at a destination address, typically a web page. In this context, web content and web applications are designed to fill the entire web page. It is known to divide the web content into different regions of a single web page. For example, personalized web pages can be specified, such that a user views a variety of content sources in a single page, such as stock information, weather information, and sports information, which is aggregated at the server that delivers the web page to the

user, who then views the aggregated content in a single web page. Observe that even when disparate content is aggregated, in this manner, it is reassembled into a full web page and is served through a full-screen browser. Web content and application developers therefore have limited control over the user experience: content is typically trapped within the frame of the browser. A developer's only alternative to engaging a user page-by-page in a browser is to develop, distribute, and support custom client software. In the Web browser scenario, it is the content provider, not the user that aggregates the information that is viewed by the user. Thus, the user is not in a position to separately aggregate the content at a client computer, instead the user is constrained to view the content that has been delivered in the manner provided by the server computer hosting the web page. There is a growing desire for individual users to fully control the aggregation and presentation of content and web applications that appears on a client computer.

(04) A user who wishes to view multiple web pages or applications can open multiple instances of a browser. However, the user will not be able to view each "full-screen" page at the same time. Instead, the user must adjust the windows corresponding to each browser instance and view only part of each page. The information appearing in each browser is not designed for viewing in this manner. Thus, the user cannot create an optimized display of content from multiple sources.

(05) Currently, content providers and end users have limited tools to alter the browser in which content appears. That is, the controls associated with a browser are not fully configurable. Thus, the vendor of a browser is in a position to brand the browser and regulate the controls associated with the browser. There is a growing desire for content providers to not only fill a browser with their content, but to also fully brand and control the

frame in which the content appears. Further, in some instances, content providers desire to limit the controls associated with a browser or viewer, so that a user is more inclined to view a single set of content, for example, by having limited access to previously viewed content.

(06) At the present time, it is relatively difficult to trace the content viewing activity of a client computer. In other words, it is difficult to identify the type of content that a particular user of a client computer favors. Consequently, there are limited tools 25 available to provide a user with tailored information that would be of particular interest to the user.

(07) In view of the foregoing, there is a need in the art to provide a technique for accessing multiple instances of distributable computer readable media in their entirety simultaneously, where these instances are typically smaller than the full pages used in 30 current web pages and web applications. There is a further need for providing the user with flexibility in selecting, collecting, relating and viewing such computer readable media, and for giving the media provider flexibility in directing media to a specific user and controlling the framework in which media is presented. Finally, there is a need to gather more accurate information regarding the type of content that a user enjoys, so that the user can be automatically provided with this content.

SUMMARY OF THE INVENTION

(08) The invention includes a method of presenting distributable computer readable media to a user in response to a user request. The method comprises the steps of identifying a definition of a Networked Information Monitor (NIM). A NIM frame is defined for the

NIM using the definition. Content is then retrieved for the NIM. Then, the content is placed in a NIM viewer defined by the frame.

(09) The invention also includes a method of altering a Networked Information Monitor (NIM). The method includes the step of receiving a message at a NIM. The message specifies a configurable feature of the NUM. The NIM is altered in accordance with the configurable feature of the message.

(10) The apparatus of the invention includes a computer readable memory to direct a computer to function in a specific manner. The computer readable memory includes a first executable module to identify a definition of a Networked Information Monitor (NIM). A second executable module defines a NIM frame for the NIM using the definition. A third executable module retrieves content for the NIM. A fourth executable module places the content in a NUM viewer defined by the frame.

(11) The apparatus of the invention further includes a computer readable memory with a first executable module to receive a Networked Information Monitor (NIM) message. The NIM message specifies a configurable feature of a NIM. A second executable module alters the NIM in accordance with the configurable feature of the NIM message.

BRIEF DESCRIPTION OF THE DRAWINGS

(12) FIG. 1 illustrates a system for processing distributable computer readable media in accordance with one embodiment of the present invention;

(13) FIG. 2 shows exemplary processing steps used to initiate an instance of a home networked information monitor (NIM) using the apparatus of FIG. 1;

(14) FIG. 3A illustrates a screen logo in accordance with one embodiment of the present invention;

(15) FIG. 3B illustrates a login construct in accordance with one embodiment of the present invention;

(16) FIG. 4 illustrates a set of visual manifestations corresponding to a collection of NIMs, including a home NIM;

(17) FIG. 5 illustrates a visual manifestation of a home NIM as well as a visual manifestation of a reference NIM that references additional NIMs;

(18) FIG. 6 illustrates how a NIM, referenced by the reference NIM of FIG. 5, is added to a processed user profile in response to a designated keyboard entry sequence or mouse click;

(19) FIG. 7 illustrates how a NIM is shared with other users in accordance with an embodiment of the present invention;

(20) FIGS. 8A and 8B illustrate screen panels that facilitate the collection of the description of a set of designated NIMs into a pack;

(21) FIG. 9A illustrates how the relative position of visual manifestations that correspond to NIMs remains fixed when the visual manifestations are within a predetermined distance of each other;

(22) FIG. 9B illustrates a representative pack in accordance with the present 20 invention;

(23) FIGS. 10A, 10B and 10C illustrate how a set of visual manifestations corresponding to a collection of NIMs is aligned against a boundary when a user selects the visual manifestations and pushes them against the boundary;

(24) FIG. 11 is a flow chart of the steps taken by a client to give a user access to a NIM where the user "collects" the NIM, in accordance with one embodiment of the invention;

(25) FIG. 12 is a flow chart of the steps taken by a client to present a NIM to a user, where the NIM has been "collected" previously by the user, in accordance with one embodiment of the invention;

(26) FIG. 13 illustrates a data structure for a NIM definition, stored in the NIM application server's template database or user profile database;

(27) FIG. 14 illustrates NMA message routing between NIMs and the message interface in the client parser application;

(28) FIG. 15 is a diagrammatic illustration of an embodiment of a NIM Management Module utilized in accordance with an embodiment of the invention;

(29) FIG. 16 is a diagrammatic illustration of an embodiment of the NIM Templates database utilized in accordance with an embodiment of the invention;

(30) FIG. 17 is an illustration of a main NIMIndex Web page used in accordance with an embodiment of the invention;

(31) FIG. 18 is an illustration of a single NIMIndex category used in accordance with an embodiment of the invention;

(32) FIG. 19 is an illustration of a full description of NIM content provided in accordance with an embodiment of the invention;

(33) FIG. 20 is an illustration of a Web page displayed to the user once the user has clicked to collect the NIM;

(34) FIG. 21 is an illustration of the main home NIM graphical user interface used in accordance with an embodiment of the invention;

(35) FIG. 22 is an illustration of a "Get New NIM" graphical user interface that may be used in accordance with an embodiment of the invention;

(36) FIG. 23 is an illustration of a "More NIMs" graphical user interface representative of an embodiment of the invention;

(37) FIG. 24 is a diagrammatic illustration of the ShareLink database used in accordance with an embodiment of the invention;

(38) FIG. 25 is an illustration of a Share NIM's graphical user interface according to an embodiment of the invention;

(39) FIG. 26 is an illustration of the main DevZone Web page utilized in accordance with an embodiment of the invention;

(40) FIG. 27 is a partial view of a NIM modification web page utilized in accordance with an embodiment of the invention;

(41) FIGS. 28A to 28D are graphical user interfaces of development NIMs (DevNIMs) utilized in accordance with an embodiment of the invention;

(42) FIGS. 29A and 29B are illustrations of Administrative Zone (AdminZone) Web pages utilized in accordance with an embodiment of the invention;

(43) FIGS. 30A and 30B are also illustrations of Administrative Zone (AdminZone) Web pages utilized in accordance with an embodiment of the invention;

(44) FIGS. 31A and 31B are further illustrations of Administrative Zone (AdminZone) Web pages utilized in accordance with an embodiment of the invention;

(45) FIG. 32 illustrates an embodiment of an event log that may be used in accordance with an embodiment of the invention;

(46) FIG. 33 illustrates the tracking of events in an event log module in accordance with an embodiment of the invention;

(47) FIG. 34 illustrates a statistics database that may be used in accordance with an embodiment of the invention;

(48) FIG. 35 illustrates a statistical analysis module and a content analysis module that may be used in accordance with an embodiment of the invention;

(49) FIG. 36 illustrates a content database that may be used in accordance with an embodiment of the invention; and

(50) FIG. 37 illustrates a user account database that may be used in accordance with an embodiment of the invention.

DETAILED DESCRIPTION OF THE INVENTION

(51) The present invention discloses a technology that is capable of processing distributable computer readable media. Distributable computer readable media includes, but is not limited to, standard web content, such as HTML, dHTML, images, imbedded ActiveX and Java applications, JavaScript, CSS, Pen scripts, Streaming Media, and/or Flash. The present invention is advantageous relative to prior art systems and methods because it provides improved mechanisms for simultaneously interacting with several independent sources of distributable computer readable media, collecting references to such media, and sharing such references with other users. The ac disclosed technology is further advantageous because it provides improved systems and methods for on screen management of distributable computer readable media.

(52) In the system and method of the present invention, a user logs into a server by providing a login identifier to a login construct. The login identifier is used by the server to obtain an unprocessed user profile that corresponds to the user. The unprocessed user profile is

processed by the server to generate a processed user profile. Advantageously, this processing step allows for up-to-date refinement of the user profile. Up-to-date refinements include, for example, the addition of advertisements directed to the user based on one or more characteristics in the user profile. The processed user profile is delivered to the client computer associated with the user. The processed user profile includes references to the networked information monitors (NIMs). As used herein, the term networked information monitor or NIM refers to a fully configurable frame with one or more controls; the frame through which content is optionally presented. The fully configurable frame utilized in accordance with the invention stands in contrast to present web browsers, which are branded by the browser vendor and which have limited means by which to alter the controls associated with the browser.

(53) Attention is initially directed toward the home NIM of the invention, which coordinates the activities of all other NIMs that are accessed by a user. The home NIM facilitates much of the technology of the present invention, including the ability to simultaneously review multiple sources of distributable computer readable media as well as to package and distribute such media.

(54) FIG. 1 is a general illustration of a system in accordance with one embodiment of the present invention. In FIG. 1, a network 10 is operated in accordance with the present invention. Network 10 includes at least one user or client computer 20, at least one server computer of class 50, and optionally one or more server computers of class 82. User computer 20 as well as server computers of class 20 50 and 82 are each connected by transmission channel 44, which is any wired or wireless transmission channel.

(55) User computer 20 is any device that includes a Central Processing Unit (CPU) 24 connected to a random access memory 30, a network connection 28, and one or more user input/output ("i/o") devices 40 including output means 42. Output means 42 is any device capable of communicating with a user and includes, for example, a video monitor, a liquid crystal display, voice user interfaces, and/or integrated graphic means such as mini-displays present in web-phones. Typically, user computer 20 includes a main non-volatile storage unit 22, preferably a hard disk drive, for storing software and data. Further, user computer 20 includes one or more internal buses 26 for interconnecting the aforementioned elements. In a typical embodiment, memory 30 includes an operating system 32 for managing files and programs associated with user computer 20. In some embodiments, operating system 32 includes a registry 34 that has one or more references to specified locations in system 10. The exemplary memory 30 of FIG. 1 further includes a web browser 36 for viewing web content and a client parser application 38 for facilitating low level functionality, such as login and logout procedures, in accordance with the present invention. In some embodiments, client parser application 38 uses the one or more references in registry 34 to obtain a login construct from server 50. In various embodiments, in accordance with the present invention, client parser application 38 runs in conjunction with one or more software modules, such as an event log module 98, which tracks user activity, a message interface module 106, which serves as a communication interface between the client parser application 38 and web server 58 and/or external web servers, a home NIM 108, which references one or more NIMs 110, and a visual management system 114 which regulates the characteristics of visual manifestations of NIMs 108 and 110 when displayed on output device 42. Furthermore, in some

embodiments, client parser application 38 runs in conjunction with instances of web browser 36 as well as web server 58 as detailed below.

(56) Server computer 50 includes standard server components, including a network connection device 46, a CPU 52, a main non-volatile storage unit 54, and a random access memory 56. Further, server computer 50 includes one or more internal buses 48 for interconnecting the aforementioned elements. Memory 56 stores a set of computer programs, modules and data to implement the processing associated with the present invention.

(57) The embodiment of memory 56 illustrated in FIG. 1 includes a web server 58 for processing requests received from client computer 20. Web server 58 has many components, including a variety of modules and data structures to assist users that want to log into system 10. Namely, login module 60 handles an entry request from a client computer 20 and accepts a login identifier that corresponds to a user from client computer 20. Login constructor 62 generates a login construct in response to a call for a login construct and transfers the login construct to client 20. Login constructor 62 dynamically generates login constructs using updated login NIM content 64. Login validation module 66 works in conjunction with user profile database manager 100 to identify an unprocessed user profile, which is associated with a user provided login, in user profile database 76. If an unprocessed user profile corresponding to the user provided login does not exist in user profile database 76, login validation module 66 associates a new unprocessed user profile with the login identifier.

(58) When an unprocessed user profile is identified by login validation module 66, it is processed by user profile processor module 68 to produce a processed profile. It will be

appreciated that the services of user profile processor module 68 are highly advantageous because they allow for last minute user profile revisions. Such revisions include, for example, the addition or customization of NIMs referenced by the user profile, and/or server redirect information that is a function of current server load in system 10. Importantly, the processed user profile includes a reference to a home NIM. The home NIM is capable of accessing each of the NIMs that are represented in the processed user profile. Memory 56 further includes delivery module 70, which coordinates the delivery of portions of the home NIM to the client based on parameters specified in the processed user profile.

(59) Once a user has successfully logged into system 10, request server module 72 handles requests for specified NIMs from client 20. When such a request is received, request server module 72 routes the request to an address that corresponds to the specified NIM and transmits the specified NIM to client 20. One class of specified networked information handled by request server module 72 is requests for NIMs. When such a request is received, request server module 72 searches NIM templates database 74 for the specified NIM. NIM templates database 74 includes a large number of NIM templates. Each NIM template defines the characteristics of a specific NIM, including fully configurable frame characteristics, viewer and control characteristics, and NIM content references.

(60) The web server 58 illustrated in FIG. 1 further includes additional modules 102 to handle specialized features of the present invention. For example, one embodiment of the present invention provides a mechanism that allows users to distribute NIMs to each other. In such embodiments, a special server module 102 provides instructions for storing the NIMs, which are to be distributed, in sharelink database 78. Advantageously, NIMs that are distributed to other users are customizable. A user can, for example, resize and position a

particular NIM prior to sharing it with another user. Indeed, it is possible, in such embodiments, for a user to arrange a series of NIMs in a unique arrangement and then distribute the collection of NIMs in the designated NIMs in the designed arrangement. As an illustration, a user arranges a first NIM that represents a scrolling stock ticker at the bottom of an output means, such as a computer screen, a second NIM that tracks the NASDAQ top ten most heavily traded stocks in the upper left corner of the output means, and a third NIM that tracks headline news on the upper right hand corner of the output means. Then, the user distributes the three NIMs in this customized arrangement to other users. Observe that in this example a user of a client computer is aggregating separate sets of information in different NIMs. This stands in contrast to prior art approaches where a web server running on a server computer aggregates information in a single page.

(61) System 10 is highly scalable and thus supports a large number of users. This scalability stems from the fact that the server 50 is delivering the definition associated with a NIM. The content displayed in the NIM may be located on a separate computer.

(62) Memory 56 provides a statistical analysis module 104 for tracking key events associated with users. This information is stored in statistics database 80. The information collected by statistical analysis module 104 is used for a wide variety of purposes, including server load optimization and directed advertising, as discussed below. As described below, the statistical information gathered in accordance with the invention includes fully traced events defining the type of content and the duration over which all content is viewed by a user. This type of comprehensive information is not available using present techniques.

(63) Much of the distributable computer readable media that is available for processing is stored as content elements 94 on server 82. Server 82 is a standard web server that includes components such as a network connection device 88, a CPU 86, a main non-volatile storage unit 84, a random access memory (RAM) 92, and one or more internal buses 90 for interconnecting the aforementioned elements. RAM 92 includes some of the content elements 94 stored by server 82. Other content elements 94 are stored in storage unit 84. In some embodiments, a single web server 58 is capable of directly accessing content elements 94 located on one or more servers 82. In other embodiments, each server 82 has a resident web server module that works in conjunction with server 50 to identify, optionally dynamically generate, and serve content elements 94 upon demand.

(64) Now that general architecture of a system in ac dance with the present invention has been disclosed, attention turns to FIG. 2, which discloses a method for logging into system 10 (FIG. 1). In the first processing step shown in FIG. 2 (202), a user initiates a session on system 10 by requesting the global login script "session_config." It will be appreciated that the term "session_config" merely provides an illustrative name for the global login script and that the technology of the present invention is by no means limited to this name or the script described.

(65) The request for "session_config" originates on client 20 and is sent to server 50 where it is processed by login module 60 of web server 58. Upon receiving request 202, login module 60 creates a "session_config" global login script (204). Processing step 204 is advantageous relative to systems that have static global login scripts because it allows for the incorporation of highly variable information. This highly variable information includes, for example, system settings such as up-to-date server redirect information, server content

address changes, directed advertisements, and messages. An exemplary "session_config" is found in Example 1 below. Each line of data has an associated numeral. The remaining text in the example describes the purpose of select data.

(66) EXAMPLE 1

(67) Version tag that identifies the latest home NIM version

(1) </SESSION_CONFIG VERSION="alpha:3"

(68) Upgrade event that is sourced when home NIM version is outdated

(2) <UPGRADE=http://www.NIM.com /QuickOpen.exe>

(69) LOCATION OF SERVER-SUPPORTED FUNCTIONALITY Default base address for server supported functionality

(3) <METHODS BASEURL=http://neo.NIM.com /servlet/NIMServer/>

(4) <ADD_USER URL="addUser/>

(5) <GET_USER URL="getUser"/>

(6) <SET_USER URL="setUser/>

(7) <GET_SES_CONFIG URL=http://www.NIM.com /home_NIM/s_cnfg.xml"/>

(70) Server-based functionality for setting password

(8) <SET_PASSWORD URL="setPassword"/>

(71) List of all publically available NIMs

(9) <GET_MASTER_NIM_LIST URL="getMasterNIMList"/>

(10) <GET_ALL_CONFIG URL="getAllConfig"/>

(11) <SET_ALL_CONFIG URL="setAllConfig/>

(12) <SET_ALL_STATS URL="setAllStats"/>

(13) <GET_NIM_TEMPLATE URL="get_NIM_Template"/>

(72) Location of server-side NIM and pack sharing functionality

(14) <ADD_SHARE URL="addShare"/>

(15) <GET_SHARE URL="getShare"/>

(16) <AUTH_TEST URL="DOeCHO?AUTH=TRUE"/>

(73) Redirect information

(17) <DO_REDIR URL="doRedir"/>

(18) </METHODS>

(74) Flexible content layer that defines default NIM frame appearance, including the default appearance of the frame of a home NIM

(19) <FRAMES>

(75) Default NIM frame appearance

(20) <NIM>

(21) <IMAGES BASEURL=

http://www.NIM.com /home_NIM/NIM_FrameImages/>

(22) </NIM>

(76) Default home NIM frame appearance

(23) <HOME_NIM>

(24) <IMAGES BASEURL=

http://www.NIM.com /home_NIMImages/>

(25) </HOME_NIM>

(26) </FRAMES>

(77) Location of system NIM templates

(27) <NIMs>

(28) <ADD TEMPLATE="http://www.NIM.com / . . . /add_NIM_XML.xml"/>

(29) <HELP TEMPLATE="http://www.NIM.com / . . . /help_NIM_XML.xml"/

(30) <LOGIN TEMPLATE="http://www.NIM.com / . . . /login2.xml"/

(31) </NIMs>

(32) </SESSION_CONFIG>

(78) Line 1 of the exemplary "session_config" of Example 1 provides the version tag for the expected version of the home dot system that corresponds to the "session_config" script. In one embodiment, client parser application 38 determines whether it is up-to-date using the information in line 1. If client parser application 38 determines that it is outdated, an upgrade request is made in accordance with the instructions provided by the UPGRADE flag of line 2. In one embodiment, the UPGRADE flag in line 2 of Example 1 describes the location of an executable program, one of skill in the art will appreciate that this flag can in fact reference any form of instruction, including a flat file, a web page, a script, a symbol, or an address.

(79) Lines 3 through 18 in Example 1 define the functionality that is provided by a server, such as server 50. For example, line 8 of Example 1 provides the location of a set of instructions that are called when a user requests a password change. Furthermore, line 9 of Example 1 provides the location of master list of NIMs that are publically available. Lines 14 and 15 of Example 1 provide the location of specialized server-side functionality that allows users to share data such as NIM definitions.

(80) Lines 19 through 26 of Example 1 define where the default appearance of a NIM and a home NIM are found within system 10. Lines 27 through 31 define a collection of system NIMs. A system NIM is any type of NIM that is to be distributed to each user of system 10.

In some embodiments, system NIMs are used to provide a core functionality. In Example 1, line 28 defines a NIM that provides users with a convenient mechanism for collecting additional NIMs. Line 29 defines the location of a NIM that is invoked when the user presses a help button associated with a home NIM. Finally, line 30 defines the location of a NIM that is used to log into system 10.

(81) Returning to FIG. 2, once login module 60 has created "session_config," it is sent back to requesting client 20 (206). When a "session_config" is received by client 20, client parser application 38 parses the global login script in order to identify a reference to a login constructor 62 (208). Login constructor 62 is a server-based module that generates a construct that allows a user to log into system 10. When client parser application 38 locates the reference to login constructor 62 in "session_config," a request for a login construct is directed to the identified reference (210). In Example 1 above, the reference to the login construct is provided in line 30. On line 30, the global variable "LOGIN TEMPLATE" is assigned the URL address "http://www.NIM.com . . . /login2.xml." Client parser application uses the URL assigned to the global variable "LOGIN TEMPLATE" to make a request for a login constructor 62 that is directed to this URL. When login constructor 62 receives a request for a login construct, it generates a login construct (212).

(82) Login construct 148 (FIG. 3B) illustrates a type of login construct that is generated in one embodiment of the present invention during processing step 212. Before the login construct is executed on client 20, a schematic such as logo 146 (FIG. 3A) is displayed on output means 42. As illustrated in FIG. 1, login constructor 62 is a component of web server 58. However, there is no requirement that login constructor 62 be a component of web server 58. In fact, login constructor 62 is a standalone software program in some

embodiments of the present invention whereas in other embodiments, login constructor 62 is merely a script, such as a PERL script, that is processed by an interpreter program native to server 50. In still other embodiments, login constructor 62 is merely a simple flat file that includes a set of instructions that are interpretable by client parser application 38. In such embodiments, login constructor 62 is the login construct. In embodiments in which a login construct is dynamically generated, it is possible to introduce last minute changes in the login construct. Thus, an advantage of the exemplary login process shown in FIG. 2 is that there are multiple stages in which updated information is used to customize the login process based on the environmental variables.

(83) Once a login construct has been prepared by login constructor 62, it is transferred back to client 20 (214) (FIG. 2) and executed in conjunction with client parser application 38 (216). The login constructs of the present invention are a form of NIM. Therefore, one function of processing step 216 is to obtain the login NIM content 64 (FIG. 1) specified by the login construct from server 64. In login construct 148, for example, the login NIM content includes the shape and functionality of "Exit button 160," message 150, the shape and functionality of "New user" button 152, the functionality of "Forgot it?" button 154, and login panel 156. When processing step 216 is completed, the user uses the login construct to provide a login identifier (218).

(84) In FIG. 3B, a user has provided the login identifier "Galliani." The definition of login identifier as used in the present invention is to be broadly construed. In some embodiments, login identifiers include a unique name and a corresponding password. In other embodiments, a login identifier does not have a password. This is particularly the case when the user is a guest or a new user and there is no user profile associated with the user.

(85) Working in conjunction with client parser application 38, the login construct accepts the user login and sends it to server 50 for validation (218). As illustrated in the exemplary system of FIG. 1, web server 58 includes a login validation module 66 to verify the login identifier provided by user (220). Typically, processing step 220 involves a look-up operation in which the login identifier is used to query user profile database 76 for an unprocessed or raw user profile that matches the login identifier. In embodiments that include a password, validation step 220 includes a password verification step. Successful completion of processing step requires entry of a valid login identifier sequence in processing step 218. When processing step 220 has been successfully completed, the raw or unprocessed user profile corresponding to the login identifier is obtained from user profile database 76 (FIG. 1) (222) and is processed by user profile process module 68 to produce a processed or finalized user profile that is delivered to client 26 (226). In sonic embodiments, a user profile 76 includes user contact information, such as the name, address, telephone number and email address of a user. Additionally, some embodiments of system 10 provide different types of access privileges. For example one embodiment of the present invention includes developer access privileges, administration access privilege, and general user access privileges. In such embodiments, the access privileges that have been granted to a user are stored in the user profile 76 associated with the user.

(86) The processed user profile includes a reference to each NIM in system 10 that is associated with the login identifier provided in processing step 216. One of the NIMs referenced by the processed user profile is the home NIM that corresponds to the login identifier provided in processing step 216. When executed in conjunction with client parser application 38 in processing step 226, the home NIM provides a mechanism for accessing

each of the NIMs referenced by the processed user profile. Like the login construct, the home NIM includes several components, including pull down menus and screen manipulation functionality. The reference to the home NIM in the processed user profile includes the system 10 address of each of these components. Therefore, in one embodiment, construction of the home NIM in processing step 226 involves one or more requests to server 50 and/or server 82 for content (228) that is then rendered (230) in accordance with the home NIM description provided in the processed user profile. In some embodiments, the home NIM is distinct from other NIMs in the sense that a large proportion of the home NIM in such embodiments is pre-compiled. Such embodiments are advantageous because some of the functionality provided by the home NIM requires substantial client 30 processing resources. Therefore, to minimize such processing resource requirements, many aspects of the home NIM are pre-compiled in some embodiments. In other embodiments, however, the home NIM has a structure that is substantially the same as a regular NIM. In such embodiments, simple script commands are used to identify the NIM as a home NIM.

(87) Upon completion of processing step 230, the user is granted access to all of the technologies of the present invention, including the ability to view multiple NIMs simultaneously, collect new NIMs, customize NIMs, and share customized NIMs with other users. An exemplary processed user profile is provided in Example 2. Once again, each line of data is identified with a numeral, while the remaining text in the example describes select data. In some embodiments, the user is granted specific privileges and the extent to which the user is granted access to system 10 is regulated by the types of privileges that have been granted to the user.

(88) EXAMPLE 2

(1) SAMPLE PROCESSED USER PROFILE

(2) <ALL CONFIG>

(89) NIMs AND PACKS THAT CORRESPOND TO THE USER

(3) USER

(90) NIM definition 1

(4) <NIM DOMAIN="ZDNet" GLOBALID="1" KND="news"

(5) <FRAME BACKGROUNDCOLOR=#FFFF00" COLLAPSED="FALSE"

(6) FIXHEIGHT="TRUE" FIXWIDTH="TRUE" NAME="ZDNet Breaking News"

(7) PIXELHEIGHT="275" PIXELWIDTH="235" X="RIGHT" Y="TOP">

(8) <TITLE COLOR=#000000" JUSTIFY="RIGHT" TEXT=" "/>

(9) <TITLEBARIMAGE DOWN=http://www.NIM.com / . . . /feed/titlebar.gif

(10) HOVER=URL address to a first GIF file <param 1> . . . <param N>

(11) INACTIVE=URL address to a second GIF file <param 1> . . . <param 2>

(12) NORMAL=URL address to a third GIF file <param 1> . . . <param 2>

(13) <BOTTOMBARIMAGE DOWN=

http://www.NIM.com / . . . /feed/bottombar.gif

(15) HOVER=URL address to a fourth GIF file <param 1> . . . <param 2>

(16) INACTIVE=URL address to a fifth GIF file <param 1> . . . <param 2>

(17) NORMAL=URL address to a sixth GIF file <param 1> . . . <param 2>

(18) </FRAME>

(19) <MENU/>

(20) <CONTROL_LAYOUT HEIGHT="1" HEIGHTSCALES="TRUE" WIDTH="1"

(21) WIDTHSCALES="TRUE"> <CONTROL CLASS="Browser" HEIGHT="1"

(22) ID="1" KIND="A" LEFT="0" TOP="0"

(23) URL=http://www.mandala.com /cgl/zdnet/zdfeedl.cgi WIDTH="1"/>

(24) </CONTROL_LAYOUT>

(25) <CATEGORIES/>

(26) <EVENTS/>

(27) </NIM>

(91) NIM definition 2

(28) <NIM DOMAIN=NIM DOMAIN 2 GLOBALID=`2`

(29) </NIM>

(92) NIM definition N

(30) <NIM DOMAIN=NIM DOMAIN 2 GLOBALID=`N`

(31) </NIM>

(93) Pack definition 1

(32) <PRESET TITLE="New DotPack">

(33) <NIM GLOBALID="1" X="RIGHT" Y="TOP"/>

(34) <NIM GLOBALID="2" X=RIGHT Y="320"/>

(35) </PRESET>

(36) </SHARE>

(94) Last state of the home NIM

(37) <LASTSTATE>

(38) <PRESET TITLE=" ">

(39) <NIM GLOBALID="1" X="RIGHT" Y="TOP"/>

(40) <NIM GLOBALID="2" X=RIGHT Y="280"/>

(41) </PRESET>

(42) <HOMENIM COLLAPSED="FALSE" HEIGHT="134" X=616" Y="109"/>

(43) </LASTSTATE>

(44) </ALL_CONFIG>

(95) Example 2 describes a representative processed user profile in accordance with the present invention. In general, a processed user profile includes three major components: (i) a definition of each NIM associated with the user, (ii) a description of each pack associated with the user and, (iii) the last state of each home NIM associated with a user. In Example 2,

the definition of each NIM associated with the user is found on lines 4 through 31. Specifically, lines 4 through 31 describe NIM definitions I through N. In Example 2 there is only one pack associated with the user. This pack, entitled "New DotPack," is found on lines 32 through 35 of Example 2. The final major component of the processed user profile found in Example 2 is the last state of the home NIM, which is defined on lines 37 through 43. This code stores the last state of the home NIM. Such last state information includes whether the home NIM was collapsed, and the position of the home NIM on the screen.

(96) When the user wishes to log out of system 10, the processed user profile is transferred from client 20 to server 50. When web server 58 receives the processed user profile, it passes the processed user profile to user profile database manager 100. User profile database manager 100 stores the processed user profile as the unprocessed user profile 76 corresponding to the user. In some embodiments, such a storage operation involves a conversion process. For example, advertisements or specific system NIM definitions are stripped from the processed user profile in order to convert the processed user profile to the unprocessed user profile 76 that corresponds to the user. In some embodiments, the processed user profile is periodically transferred, in its entirety or incrementally, from client 20 to server 50 and saved in the manner described in the log out procedure above. Such timed periodic or event based backup procedures are possible because NIM definitions are efficiently described, thus the absolute size of a processed user profile remains relatively small. Accordingly, timed backups of a processed user profile to user profile database 76 are possible without extensive use of system 10 bandwidth or server 50 resources.

(97) At this stage, a system (FIG. 1) and a login procedure (FIG. 2) in accordance with the present invention has been disclosed. Although the system and login procedure was discussed using an example where only one home NIM was associated with a user, it will be appreciated that, in some embodiments, any number of distinctly different home NIMs are associated with a user. Furthermore, a user can simultaneously execute multiple instances of a particular home NIM on client 20 or, indeed, any number of different home NIMs. In one embodiment, a developer or merchant provides a user with a highly customized home NIM that provides specialized functionality. In such embodiments, the user collects the home NIMs and, therefore, a processed user profile includes a description of more than one home NIM.

(98) Attention now turns to some of the advantages and features of the present invention. In FIG. 4, a visual manifestation of the home NIM 162 is illustrated. One advantage of the home NIM, which is an advantage that is common to NIMs in general, is that the content of the NIM is not trapped in a third party viewer. In fact, the home NIM definition regulates the actual appearance of the home NIM. The home NIM definition is formed by general parameters and commands found in "session_config" as well as customized parameters and commands in the processed user profile. The division of the home NIM definition between a system level file and a user level file represents a balance in the tension between the need for a system 10 host to insure a consistent level of quality, through the proper implementation of general parameters and commands, and the desire of each user to create highly customized home NIMs. Lines 20 through 22 of Example 1 provide an example of general parameters that are defined in "session_config." Lines 23 through 25 define the source location of home NIM frame images. In home NIM 162 (FIG. 4), such home NIM

frame images include the image used to represent buttons 164 through 174, and menu tabs 1 d 178. Furthermore, lines 20 through 22 of Example 1 define the location of other images that are used to construct default NIMs. An example of user initiated home NIM customization is found in lines 33 through 39 of Example 2, which define a "LASTSTATE" definition for the home NIM, including the dimensions of the visual manifestation corresponding to the home NIM on line 38 (HEIGHT="134" X="616" Y="109") and indicates that the home NM is not collapsed upon startup (COLLAPSED="FALSE").

(99) The visual manifestation of home NIM 162 illustrates additional benefits and features of a home NIM in accordance with the present invention. When a user selects tab 176, a list of the NIMs that are present in the processed user profile associated with the user is displayed in viewer 180. As disclosed in more detail below, a user has the option to associate a collection of NIMs into an object termed a "pack". The pack references some subset of the NIMs associated with a user as well as associated state information. This arrangement includes, for example, whether a visual manifestation corresponding to each MM is displayed on output means 42 or not, the dimensions of each visual manifestation, and the position of each visual manifestation. The name of each pack is stored in the processed user profile. A user reviews packs associated with the user by selecting tab 178 (FIG. 4). In FIG. 4, the user only has one pack, "Customized DotPack" 182. When the user selects pack 182, each NIM in the pack is restored in accordance with the state information stored in the pack definition.

(100) In total, FIG. 4 represents a typical visual experience provided by one embodiment of the present invention. In addition to home NIM 162, visual manifestations 184 and 186, corresponding to two additional NIMs in the processed user profile, are displayed. Visual

manifestation 184 provides functionality that allows a user to manage an address book, schedule appointments, or create groups and plan activities. Visual manifestation 186 represents a NIM that provides time and date information.

(101) FIG. 5 shows the visual manifestation of home NIM 162 of FIG. 4 with tab 176 selected. Accordingly, each of the NIMs in the processed user profile associated with the user is listed in list 188. The user can activate any of the listed NIMs by clicking on the NIM name. In addition to the NIMs in list 188, home NIM 162 includes core NIMs that are defined in "session_config." In the "session_config" of Example 1, cores are found on lines 28 and 29. Specifically, line 28 provides the address of an XML-based definition for the add template functionality associated with button 172 in FIGS. 4 and 5, and line 29 provides the address of an XML-based definition for the help template functionality associated with button 174 in FIGS. 4 and 5.

(102) Importantly, the user can categorize NIMs using filter 190. Categories include such topics as sports, personal, weather, etc. Furthermore, the user can add NIMs to the processed user profile associated with the user as well as delete NIMs. There are a variety of mechanisms that enable a user to add a NIM to the processed user profile. One mechanism is to receive links to NIMs from other users of system 10 (FIG. 1), as disclosed below. Another mechanism is to toggle button 172 in order to activate a visual manifestation associated with NIM 192 (FIG. 5).

(103) NIM 192 provides a system that enables users to add select NIMs to their user profile with a single click or keystroke sequence. NIM 192 includes tab 194 that allows the user to select premiere NIMs and a general tab 196 that allows the user to review a general catalog of NIMs that is present in NIM templates database 74 (FIG. 1). In one embodiment, when a

user selects a NIM in list 198 (FIG. 5), the NIM is added to list 188 and is incorporated into the processed user profile associated with the user. In this way, the user can collect NIMs of interest to the user using a single mouse click. By illustration, consider the case in which a user selects the NIM "AnyDay Calender" in list 198. In response to this selection, a definition of the NIM "AnyDay Calender" is obtained from NIM templates database 74 and is copied directly into the processed user profile associated with the user. Furthermore, the title of the selected NIM, "AnyDay Calender" is added to list 188. Finally, a visual manifestation that corresponds to the NIM "AnyDay Calender" is displayed on output means 42 (FIG. 1). As a result, the display illustrated in FIG. 5 adopts the appearance illustrated in FIG. 6.

(104) In FIG. 6, the NIM "AnyDay Calendar" appears at the top of list 188. Furthermore, a control 101 associated with the NIM "AnyDay Calendar" in list 188 is filled, indicating that the NIM is currently active. Additionally, as illustrated in FIG. 6, a visual manifestation 103 corresponding to NIM "AnyDay Calendar" appears on the output means. The user has the ability to toggle this NIM between an inactive and active state by selecting control 101.

(105) In one embodiment, the user is provided with the option of (i) incorporating a NIM selected in list 198 into the processed user profile or (ii) transiently executing the NIM on client 20. Furthermore, when the user receives NIMs from other users, the user has the option to transiently operate the received NIMs on client 20. If the user decides to keep the transient NIMs at a later date, the user has the option to add the transient NIMs to the processed user profile at that time. Thus, in such embodiments, the user effectively has the option to "preview" NIMs before adding them to the processed user profile. This is advantageous because it reduces the chances of filling the user profile with undesirable

NIMs. Such a feature is particularly advantageous in the case of novice or inexperienced users of system 10. Furthermore, one of skill in the art will appreciate that the concept of transient NIM execution raises the possibility of executing NIMs on a client 20 during a period of time in which the user is not logged into system 10. For example, consider a NIM that is executed on a client 20 after a user initiated response to a web page advertisement presented in web browser 36. Although the user is not logged into web server 58 and therefore does not have a processed user profile resident on client 20, the user can execute the NIM on client 20 on a transient basis. Furthermore, if the user wishes to add the transiently executed NIM to the user profile 76 associated with the user, the user can log into web server 58 and then add the NIM to the processed user profile that is delivered to client 20 as a function of the log in process.

(106) Another important feature of the present invention is the ability for users to share NIMs with each other. For example, if a user wishes to share the NIM "AnyDay Calendar" that was added to list 188 in FIG. 6, the user clicks "share" button 170 (FIG. 6). In response, panel 105 is displayed (FIG. 7). Because "share" button 170 is pressed while tab 176 is active in the illustration provided by FIGS. 6 and 7, panel 105 lists each of the NIMs associated with the user. If, however, "share" button 170 is pressed while tab 178 is active rather than tab 176, panel 105 will list each of the packs associated with the user instead of each of the NIMs. Returning to the situation illustrated in FIG. 7, the user shares a NIM with other users by selecting the NIM to be shared from list 107 and then toggling button 109 "Share via email." In one embodiment, the user has the option to select multiple NIMs from list 107 using predefined keystroke operations. For example, in one embodiment, the user selects multiple NIMs by clicking on several of the NIMs in list 107 with a mouse

button while depressing the "shift" button on the keyboard. When a user decides not to share a NIM and panel 105 is displayed, the user presses cancel button 111 and panel 105 is dismissed.

(107) When a user toggles "share via email" button 109 at a time when one or more NIMs in list 107 have been selected, the definition of each selected NIM is copied from the processed user profile associated with the user into a container and the container is sent to server 50 (FIG. 1). In the embodiment shown in FIG. 1, the container is received by web server 58. Web server 58 includes instructions for routing the container to sharelink database 78 where the container is stored. When the container is stored, a unique identifier is assigned to the container. Although a large number of different mechanisms for generating a unique identifier are practiced in accordance with this aspect of the invention, in one embodiment, the unique identifier assigned to the container upon storage in sharelink database 78 can be subsequently processed to form a URL address that specifically references the container within the context of system 10. In one embodiment, after a unique identifier has been assigned to the container, an e-mail program is launched on client 20 and the user is requested to designate the recipients of the designated NIMs. Then, each recipient is provided with the unique identifier associated with the container in an e-mail message. When the recipient clicks on the unique identifier, a call is made for a copy of the associated container from sharelink database 78 and the container is delivered to the client 20 associated with the recipient.

(108) As is readily apparent upon review of FIG. 7, the user has the option to size and position the visual manifestation that corresponds to each NIM. Furthermore, by toggling controls, such as toggle button 101 (FIG. 7), the visual manifestation of a NIM is toggled between an

on state and an off state. Such functionality is highly advantageous. First, by using this functionality, the user has the option to create unique arrangements. Second, NIM developers have the ability to control the default position and size of NIMs as well, and can therefore produce an arrangement of NIMs to further specialized purposes. Finally, because the NIMs of the present invention are not trapped in third party applications that have a set of undesirable features such banner ads, the utility and overall appearance of an arrangement of NIMs is enhanced and adopts an independent value. Using the technology disclosed in the present invention, the user collects an assortment of NIMs and arranges them in a customized fashion. The user has the option to "capture" favored arrangements into constructs known as packs, which have been briefly discussed previously.

(109) FIGS. 8 and 9A illustrate the formation of a pack using the arrangement of NIMs illustrated in FIG. 4. The process begins when the user toggles button 164 "Make Pack" in FIG. 4. In the embodiment illustrated by FIGS. 8 and 9A, panel 113 (FIG. 8) is displayed when the user toggles button 164 (FIG. 4). Panel 113 advises the user to open and arrange each of the NIMs that are to be included in a pack. In the case of FIG. 4, for example, such an arrangement could include the arrangement of NIMs 184 and 186. The user indicates that specified NIMs are in a desired arrangement by selecting button 115 "Next" (FIG. 8). When button 115 is toggled, prompt 113 is terminated and prompt 117 is displayed to prompt the user for a name to associate with the designated pack. The user indicates that a name 119 has been provided for the pack by selecting "Done" button 121.

(110) In the embodiment shown in FIG. 8, the user further has the option to return to panel 113 and rearrange the specified NIMs before committing to pack creation by selecting the "Back" button 123. In the situation illustrated in FIG. 8, the user has provided the name

"New DotPack." FIG. 9A illustrates the state of the visual manifestation corresponding to home NIM 162 after the user has selected "Done" button 121 (FIG. 8). Specifically, the name "New DotPack" is added to pack list 125 and tab 178 is activated to display the user pack list rather than the user NIM list that is displayed when tab 176 is activated. Furthermore, in response to the user selection of "Done" button 121 in FIG. 8, a reference to each NIM specified by the user is collected into a pack, along with some state information, and the pack is stored in the processed user profile associated with the user. Representative state information for each NIM stored in a pack includes whether the NIM was collapsed and the position of the NIM. In some embodiments, the state information includes the dimensions of the last visual manifestation corresponding to the NIM to have been displayed on output means 42.

(111) FIG. 9 illustrates pack 139, which is delineated with a dashed box. Pack 139 includes five NIMs 133. Each NIM 133 includes two primary components, a viewer 135 for viewing content and a frame 137 for providing user functionality. Each viewer 135 provides a platform for reviewing machine readable information, such as HTML, dHTML, images, imbedded ActiveX and Java applications, JavaScript, CSS, Streaming Media, and/or Flash. Furthermore, in some embodiments, viewer 135 provides functionality for processing digitally recorded sound. Observe then that each NIM of the invention utilizes machine readable information that is easily retrieved from a specified address. If the content provider of this information desires to change the machine readable information, changes can be made and then delivered the next time that the machine readable information is addressed. This stands in contrast to prior art techniques in which updates to executable code can be relatively difficult to distribute.

(112) Each frame 137 provides functions and controls for manipulating the visual manifestation of the NIM 133 corresponding to the frame. For example, some frames 137 include a dismiss button 141. When the user clicks on dismiss button 141, the corresponding NIM 133 is dismissed. Additionally, some frames 137 include a menu button 143. When the user clicks on button 143, a menu appears. In one embodiment, the menu is a pull down menu and the menu items are defined in the corresponding NIM definition. In an abstract example, the NIM definition provides a menu definition of the format:

Menu 143-(I)-"Get more NIMs" URL

(113) where (I) refers to the N.sup.th element of the menu that is activated when menu button 143 is pressed, "Get more NIMs" is the menu element name, and URL is the event or web address that is called when the user selects the N.sup.th element of the menu associated with button 143.

(114) The developer has full control over all aspects of the appearance and functionality of NIM frame 137. Thus, a NIM developer has the ability to control, for example, the functionality located at any corner of frame 137, functionality placed along the top or bottom of the frame, or on the sides of the frame. As an illustration, frame 137-3 includes a control button 145 that allows the user to resize the visual manifestation of NIM 133-3. One of skill in the art will appreciate that the bottom row of NIM frame 137 could have any number of buttons, including a forward control, a backward control, and similar controls, each of which provides the user with distinct functionality.

(115) An advantage of the present invention is that it is possible to embed commands that affect frames 137 in the content of the computer readable media delivered to frame viewer

135. The use of embedded commands provides NIM developers with powerful tools and additional flexibility. For example, a developer can use embedded commands, such as the menu command above, to design a NIM that has a context sensitive help menu. Each time a particular event occurs, the developer changes the content of the help menu using embedded commands. For example, when a sound file is delivered to a NIM, the sound file may be attached to a set of embedded commands that provide sound manipulation functionality in the form of a pull down menu. Elements of such a pull down menu include a command for saving the sound file to disk, commands for fast forward, stop, and play, and commands for sound enhancement. After the sound file has been played, embedded commands are used in this example by the developer to reset the menu associated with button 143 to some default state. One of skill in the art will appreciate the benefits and advantages of a frame 137 that is capable of being modified based on commands embedded in the machine readable media delivered to the corresponding viewer 135. The developer can use presentation tools, such as adding transient help buttons, resizing the visual manifestation of the NIM, changing the frame border color, changing the title of the frame, and changing the frame border patterns, to create a more effective application.

(116) One of the advantages of the disclosed pack system is that it provides a convenient mechanism for rapidly assembling NIMs that track diverse sources of information. Furthermore, when a useful set of NIMs is collected into a pack, the user can share the pack with other users using the same procedure previously identified for sharing one or more NIMs. It will be appreciated that in some embodiments, NIMs are commercial applications and that appropriate use of packs provides an additional dimension for application development. Therefore, in some embodiments, pack recipients are subscribers to a service

provided by a pack developer. In other embodiments, pack recipients are purchasers or licensees of packs. In still other embodiments, pack recipients receive packs developed by friends, family members, or business associates.

(117) The technology of the present invention further provides a set of NIM management tools to help a user manage displayed NIMs. In some embodiments of the present invention, these management tools are provided by visual management module 114 (FIG. 1). Two such NIM management tools are, in fact, properties that are associated with NIMs, namely magnetism and snapping. The property of magnetism is exhibited when a visual manifestation corresponding to a first NIM is dragged or moved near a visual manifestation corresponding to a second NIM. When this occurs, the two NIMs exhibit a magnetism that causes the first NIM to accelerate toward the second NIM. However, when the visual manifestations are within a predetermined distance of each other, the NIMs snap together. In one embodiment, the predetermined distance that triggers the two visual manifestations to snap together is a gap of about five pixels. While the above discussion describes the principles of snapping and magnetism based on a pair of NIMs, there is in fact no limitation on the number of NIMs that can be snapped together and furthermore, the principle of magnetism is not dependent on whether a NIM is in fact snapped to another NIM or not.

(118) An additional management tool, illustrated in FIG. 9A, provides a mechanism for selecting multiple NIMs and for moving the NIMs in a coordinated fashion. In FIG. 9A, visual manifestations 162, 184 and 186 corresponding to respective NIMs are locked together. In response, halo 127 is drawn around the selected locked visual manifestations to graphically notify the user which NIMs are locked together. As mentioned previously, the user has the option to position NIMs as a coordinated group. For example, in one

embodiment, when halo 127 is displayed and the user moves mouse arrow 129 after selecting one of the visual manifestations corresponding to a locked NIM, a target manifestation 131 is displayed to indicate to the user the position that the selected NIMs will be relocated to if the user clicks a mouse key. In some embodiments, target manifestation is a shadow image of the NIMs within halo 127 rather than the box depicted in FIG. 9.

(119) The present technology further provides additional methods for controlling visual manifestations of NIMs. For example, in one embodiment, the user has the option to select multiple NIMs by pressing a predefined key such as the keyboard "Ctrl" key, before selecting a specified NIM. While continuing to depress the control key, the user has the option to select additional NIMs and add the corresponding NIMs to a group. The user then has the option to move each of the NIMs in a single coordinated fashion as a group. Furthermore, by entering a designated keyboard or mouse sequence, the user has the option to move a single NIM even in situations where the NIM is in a group. In one embodiment in accordance with this aspect of the invention, the user clicks the visual manifestation corresponding to a locked NIM that the user wishes to move in an independent manner and the user does not click the visual manifestation when the user wishes to move the NIM in a manner that is coordinated with the other NIMs. Additional features of the present invention include the option to select rows or columns of NIMs using specialized control sequences. For example, in one embodiment of the present invention, the user selects a column of NIMs by clicking on a NIM while holding down the alphanumeric character "c" on the keyboard.

(120) It will be appreciated that one advantage of the present invention is that it is possible to display multiple NIMs and that each NIM provides a specialized visual experience. Therefore, NIM alignment tools are advantageous because they allow users to quickly make room on output means 42 for additional NIMs and/or to produce highly styled NIM arrangements. Accordingly, the present technology provides a specialized feature to rapidly align NIMs. This technology is illustrated in FIG. 10. The technology works in conjunction with the tools for selecting multiple NIMs. In FIG. 10A, the user selects the visual manifestations 147, 149 and 151 corresponding to respective NIMs using, for example, the column select feature disclosed above. Then, the user pushes the selected NIMs against boundary 153. In one embodiment, boundary 153 is the horizontal or vertical edge of output means 42. In another embodiment, all visual manifestations corresponding to NIMs are displayed in a single viewport such as a window. In such embodiments, the horizontal and vertical edges of the window each represent a boundary. FIG. 10B illustrates how visual manifestations 147, 149, and 151 are automatically aligned when they are pushed against a boundary, such as boundary 153. In some embodiments, the relative alignment between the visual manifestations is preserved even after the manifestations are moved in subsequent action by the user. The present technology further allows for the rearrangement of NIMs along a particular axis. For example, if NIMs are substantially oriented along a vertical axis as shown in FIG. 10B and the user wishes to realign the NIMs on the horizontal axis, all the user has to do is push the collection of NIMs against a horizontal border. For example, when the user pushes visual manifestations 147, 149 and 151 against border 155, the visual manifestations realign to conform to border 155 thus resulting in the view depicted in FIG. 10C.

(121) The features of the home NIM of the invention have been fully described. Attention presently turns to the architecture and operation of individual NIMs utilized in accordance with the invention.

(122) In one embodiment of the invention, after a user has logged into the system, as discussed above in connection with FIGS. 1 to 10, the user interface 40 displays the home NIM 162 as shown in FIG. 5. The home NIM typically includes a list of NIMs 188, referred to in FIG. 5 as "MyDots." These are NIMs which have been "collected" by the user. The list of collected NIMs, along with their associated definitions, is stored on the server in the user profile database 76, and downloaded from the application server 50 in response to a request from the client parser application 38. The local copy of the processed user profile is then further processed when the user collects or uses NIMs.

(123) Collected NIMs may be opened or closed by clicking on the control button next to the NIM's name or on the NIM's name itself, in list 188, and all NIMs may be closed by clicking on the "all off button` 166. The user may place NIMs into categories in a list of categories 190, which can be edited by clicking on the-Edit" button 168. New NIMs may be added to the user's collection of NIMs by clicking on the "Get" button 172.

(124) FIG. 5 also shows a NIM 192 with a list of NIMs 198, which may be previewed and/or collected by the client 20. The user may preview or collect a NIM by clicking on the associated name of the NIM, e.g., "eHOW". The steps taken to provide the NIM to a user are shown in FIG. 11. After logging in (step 240) the user clicks on the name of a NIM, and the client parser application 38 sends a request including the NIM_ID of the NIM definition, to the applications server 50 via the transmission channel 44 (step 241). Alternatively, the user may click on a NIM link before logging in, for example if the link

has been e-mailed to the user, and then, after clicking on the link, log in. In another aspect, the user could view, but not collect, a transient NIM without ever logging in.

(125) After the user is logged in and has clicked on the NIM, the applications server 50 retrieves the NIM definition from the NIM template database 74 using the NIM ID, in step 242, and provides it to the client 20 in step 243. The client 20 receives the NIM definition from the applications server 50 in step 244, and the client parser application 38 creates a frame in the display of the user interface 42 in step 245. In step 246, the client 20 requests the necessary content elements 94 stored at the URLs identified in the NIM definition from the corresponding content server 82. The content server 82 transmits the content 94 in step 247, and in step 248 the client parser application 38 places the content in the viewer, which is enclosed by the frame, allowing the user to preview the NIM. Alternatively, the client parser application 38 may simply collect the NIM, adding it to the user's processed user profile.

(126) The user may then view the NIM on the user interface display 42, and may interact with the NIM much in the same way as a user may interact with Internet content or web applications. This may change the NIM from its present, "raw" state to a used state reflecting alteration or use of the NIM by the user. For example, the user may direct the NIM to different content within the NIM if the NIM content enables the user to do so. Or, the user may provide information to the content server 82 which allows the NIM to be personalized. The user may additionally be given the option of changing the size of the frame.

(127) If the user collects the NIM, the NIM will be added to the user's list of collected NIMs such as the list 188 shown in FIG. 5. Additionally, the client parser application will add the

NIM's definition to the processed user profile, and, on logout, send the processed user profile to the application server 50. Thus, the NIM's "state" will be preserved. Alternatively, the client parser application may collect the NIM automatically, without waiting for a user command, by adding the NIM definition directly to the processed user profile.

(128) If the NIM's state has been altered by the user or by the content--if for example, the user has directed the NIM to Internet content other than the initially-displayed content, provided personalizing information, or changed the properties of the frame, or if the content itself has caused an alteration in the NIM--this alteration will be reflected in the NIM definition stored in the user profile database 76. Information which personalizes the resulting content, instead of being stored in a "cookie" on the client's hard drive, can be stored as part of the NIM definition. This advantageously permits personalization of content, such as web content that is associated with the NIM content and the user, without storing a cookie on the client 20.

(129) A user may also access a NIM which has been previously collected, and possibly altered by use as explained above. As previously described, the user profile 76 includes NIM definitions for NIMs which have been viewed and collected by each user. A screen shot showing NIMs 188, which have been previously collected by a user is shown in FIG. 5. The steps taken to provide the user with NIMs which have been previously collected are shown in FIG. 12.

(130) As discussed above, on login (step 250) the user's profile is retrieved by the client parser application 38 in the client 20 (step 251 and 252). The user's profile, stored in the user profile database 76, includes the NIM definition for each of the NIMs previously collected,

and possibly altered, by each user. The NIM definitions, as discussed above, includes the NIM frame definition and the definition of the controls for filling the viewer within the frame with content. After log in, a local copy of the processed user profile is stored on the client 20, and this copy is further processed as the user collects new NIMs, or uses new or collected NIMs such that the NIMs are altered.

(131) When the user clicks on the name of a collected NIM (step 253) the client parser application 38 creates a frame in the display 42 of the user interface 40 in step 254. At step 255, the client 20 requests the necessary content elements 94 stored at the URLs identified in the NIM definition from the corresponding content servers 82, which provide the content 94 in step 256. It will be appreciated that these URLs need not be the same as the initialization URLs in the "raw" NIM definition stored in the NIM template database 74 on applications server 50, and in fact the content servers need not be the same content servers corresponding to the initialization URLs. In step 257, the NIM parser application 38 places the content in the NIM frame, and the NIM is then fully opened.

(132) FIG. 13 illustrates a data structure for a NIM definition. As discussed above, a NIM is defined as a frame that contains a collection of controls, or functional units, such as a web rendering control or a GIF rendering control. The NIM frame surrounds a viewer, which displays the addressed content. The MM has a defined layout or arrangement of the controls, and defined initialization input data, e.g. data and URLS, for each control or element, in the NIM. NIM definitions are available to the client parser application via NIM links. The NIM links "point" to NIM definitions, which include all the information needed to build a NIM frame and fill the NIM with NIM content. Thus, NIMs links are easily collected, associated into packs, and shared by users.

(133) In one embodiment, the NIM definitions are defined using Extensible Markup Language (XML), so that the NIM as a whole--the frame and the content within the viewer--is advantageously as flexible as standard web content. NIMs are extremely flexible, because the definition of the NIM is content, rather than compiled code. The NIM definition defines the structure of the NIM, and everything that is visible in a NIM is based on standard Internet content, such as HTML, dHTML, or GIFs, and is referenced or pointed to by the NIM definition. An "application"-type NIM, such as a web calendar or web mail, may be changed by the user, by the content provider, or by other content, while advantageously avoiding the need to distribute and support a hard-coded compiled application. The definition of a NIM thus includes everything that is needed for the NIM to be rendered and filled with Internet content.

(134) As shown in the exemplary embodiment of FIG. 13, the definition of a NIM includes tags that identify the NIM 270, define and configure the NIM frame 271, specify and layout the controls 273 in the NIM viewer, and specify parameters to initialize all the NIM's components with content or data.

(135) In one embodiment, a NIM is identified by three ID strings 270: GlobalID, Domain and Kind. A GlobaliD is used when the MM definition is within a share. It is unique with respect to other NIM tags in the share. A NIM's domain is a unique label for the owning company or developer of the NIM, such as "dodots.com ." Finally, a NIM's kind, which is specified by the NIM's developer, is a helpful identifier for finding the NIM, but need not be unique. Examples of possible NIM kinds include "mp3", "scriblets," and "calculator." As discussed above, a NIM definition will typically be written in a format which facilitates sharing of data over the Internet, such as XML. An XML specification for the NIM

identification strings, for one embodiment of the invention follows. The bold text identifies NIM definition data, while the remaining text describes the data.

GLOBALID="string" Used only within <SHARE> tags. This GLOBALID must be unique with respect to other <NIM> tags in this <SHARE>.

DOMAIN="string"

(136) Unique label for the owning company of this NIM. In theory, NIMs may be limited to communicating with NIMs only from their own domain.

KIND="string"

(137) Helpful identifier for finding such a NIM from another NIM. Does not have to be unique.

(138) The NIM definition also includes the definition of a frame 271, which specifies the frame size and shape, and optionally the frame orientation and/or location on the user's screen. The space within the frame is the control space or viewer; visible controls are distributed within the control space or viewer.

(139) The NIM definition may optionally include controls for: a titlebar; a NIM menu with flexible menu entries; an exit button; and a bottombar. A typical layout for these components is: titlebar at the top of the control space, with menu on the left and exit button on the right, and the bottombar at the bottom.

(140) The titlebar component gives the user a place to grab and drag the NIM in a windowed environment. In one embodiment, it is implemented as a GIF rendering control that can be

targeted to a local or remote titlebar image. The titlebar will preferably have a fixed height and width that is a function of the NIM's width. The titlebar is preferably capable of being located at any position on the periphery of the NIM. Overlay text can also be specified to layer on top of the titlebar image. The bottombar may be implemented in a similar fashion, but typically will not include text overlay. The titlebar and bottombar may be filled in with initialization data from a fixed data file, or alternatively with Internet content from, example, an initialization URL.

(141) In one embodiment, a menu definition 271 is also included in the NIM definition. The menu includes items and actions of the NIM provider's choosing. For example, menu items may include the title "browse" associated with the action of targeting a full-screen browser or another NIM, and retrieving content for that browser or NIM from a specified address such as a URL. Logging off, or directing the NIM to another address or URL, are also possible menu action items. Menu action items that require communication of messages between the NIM and another NIM may also be provided--for example, opening another NIM, or changing the content of another NIM that is already open. Communication of messages between different parts of the system is discussed below.

(142) An XML specification for a frame, titlebar, bottombar and menu, for one embodiment of the invention follows:

<FRAME>

<TITLE>

TEXT="string"

JUSTIFY=-"LEFT"|"CENTER"|"RIGHT"

COLOR="#XXXXXX"

PIXELWIDTH="integer"

Width in pixel units. Overrides WIDTH attribute.

PIXELHEIGHT="integer"

Height in pixel units. Overrides HEIGHT attribute.

WIDTH="integer"

Width in NIM units. Default value is 1.

HEIGHT="integer"

Height in NIM units. Default value is 1.

X="integer"|"LEFT"|"CENTER"|"RIGHT"

Initial X position in screen coordinates. Default is center.

Y="integer"|"TOP"|"CENTER"|"BOTTOM"

Initial Y position in screen coordinates. Default is center.

FIXWIDTH="TRUE"|"FALSE"

Default is false.

FIXHEIGHT="TRUE"1"FALSE"

Default is false.

BACKGROUND COLOR="#XXXXXX"

Default is white.

<TITLEBARIMAGE>

JUSTIFY="LEFT"|"CENTER"|"RIGHT"

TILELEFT="integer"

TILERIGHT="integer"

NORMAL="URL"

DOWN="URL"

HOVER="URL"

INACTIVE="URL"

<BOTTOMBARIMAGE>

JUSTIFY="LEFT"|"CENTER"|"RIGHT"

TILELEFT="integer"

TILERIGHT="integer"

NORMAL="URL"

DOWN="URL"

HOVER="URL"

INACTIVE="URL"

<MENU>

Contains zero or more <ITEM> tags.

<ITEM>

TITLE="string"

TOOLTIP="string"

ICON="URL"

ID="string"

Must be unique.

<ACTION> RECIPIENT="address" MESSAGE="string"

(143) As shown in FIG. 13, the NIM definition also includes layout and definition of the controls 273. A control may be visible and render some sort of visual or text display, either static or dynamic. A control may be hidden, for example a functional element that is not necessarily visual such as a Java control. The control definition 273 includes identification of the types of controls, the layout of the controls, and initialization information. In one embodiment, NIM controls are specified and identified by class, kind and ID. Class defines the type of NIM control and is not unique. Kind is a useful identifier selected by the developer, and again is not unique. The NIM ID is unique within a user's processed profile.

(144) Different classes of controls may be used. For example, a control may be a web rendering object, which can render web content such as HTML, dHTML, images, imbedded ActiveX and Java applications, JavaScript, CSS, Streaming Media, and/or Flash. Alternatively, a control may be any object capable of rendering any kind of computer readable media, such as a GIF rendering object or an custom-designed object to display a particular kind of information. Alternatively, a control may be an object capable of processing any kind of application logic, such as a Java module. For example, an on-line brokerage firm could implement a custom stock-chart-rendering control, and define a NIM to use this control.

(145) As discussed above, the control space is located within the frame, and one or more controls can be flexibly positioned within the control space, and these controls may include a titlebar and a bottombar, as well as other controls. The location of controls is specified by the layout in the definition of the controls 273 within the NIM definition. In one embodiment, the controls are laid out or positioned within the NIM frame according to a flexible grid. In this embodiment, the NIM definition allows the control space to be

subdivided into equal vertical and horizontal units, and then for the controls to be positioned and sized within the control space.

(146) A control definition will typically include initialization data. For example, where a control is a web rendering object, the definition will include initial URLs. When the NIM is opened, the control will navigate to the initial URLs to obtain content and render the NIM. If the control is a GIF, the control could retrieve the GIF file from a content server or from the application server. A NIM definition may optionally include additional tags identifying initialization parameters for different platforms: for example, a URL is suitable for a PC, but a "P-URL" may be provided as well, pointing to content suitable for users viewing NIMs through a personal digital assistant (PDA) or similar device.

(147) Controls are typically installed on the applications server, and may be updated after installation by the applications server. The home NIM code, downloaded when the client becomes "NIM-enabled," includes the then-existing controls. Controls are updated as new controls are installed on the server or when a user requests a NIM that requires a new control. The server may then may download such updates to the client parser application, for example on log-in. The NIM framework allows any control to be positioned and initialized in a control space in a NIM, as discussed above.

(148) An XML specification for control definition and layout, in accordance with one embodiment of the invention, follows:

<CONTROL_LAYOUT> Contains zero or more <CONTROL> tags.

WIDTH="integer" Divides control space into this many evenly spaced columns. Default is

1.

HEIGHT="integer" Divides control space into this many evenly spaced rows. Default is 1.

WIDTHSCALES="TRUE"|"FALSE" Default is true.

HEIGHTSCALES="TRUE"|"FALSE" Divides control space into this many evenly spaced columns. Default is 1.

<CONTROL>

CLASS="string" Class may be "Browser," "GIF reader," or other object for rendering computer readable media.

KIND="string"

ID="string" Must be unique with other controls in this NIM.

LEFT="integer" X position of the control in container unites. Default is 0.

TOP="integer" Y position of the control in container units. Default is 0.

WIDTH="integer" Width in container units. Default is 1.

HEIGHT="integer" Height in container units. Default is 1.

URL="URL" This is read if and only if this control is of class "Browser". This is the URL to which this control navigates. Otherwise, may include address for other control content, e.g. GIF address in applications server.

<CATEGORIES> Contains zero or more <CATEGORY> tags.

<CATEGORY> Adds the NIM to this category. This is the only way categories are specified; i.e. there is no master category list.

NAME="string" This is the name of the category.

(149) A NIM definition may also optionally include home NIM categories 274. A home NIM category used by home NIM 204 is a convenient way for a user to keep track of collected NIMS. When a user adds a NIM to a category 204, the category is added, as a string element, to the categories element 274 of the NIM definition in the user profile. For example, a user may categorize a particular NIM as "entertainment," or "news," or "reference." This category will then be added to the categories element 274 of the MM definition.

(150) A NIM definition may also optionally include an events element 275, which defines actions to certain NIM events. For example, the OnClose event, when a NIM is closed, may be assigned a specific and targeted action, similar to a menu item. An XML specification

for the event element in a NIM definition, in accordance with one embodiment of the invention, follows:

<EVENTS>

<ONCLOSE>

(151) Executes this action list on close.

<ACTION_LIST>

(152) Contains zero or more <ACTION> tags.

<ACTION>

RECIPIENT="address"

MESSAGE="string"

(153) A sample NIM definition, in an XML file format in accordance with the above specification, follows:

<NIM DOMAIN="calculator" KIND="basic">

<FRAME CLASS="Standard" BACKGROUNDCOLOR="#FFFF00"

WIDTH="6" HEIGHT="4" FIXWIDTH="TRUE" FIXHEIGHT="TRUE">

<TITLE TEXT="Basic Calculator" COLOR="#000OFF" JUSTIFY="LEFT"/>

<TITLEBARIMAGE JUSTIFY="LEFT" TILELEFT="1" TILERIGHT="1"

NORMAL=" " INACTIVE=" " HOVER=" " DOWN=" "/>

<BOTTOMBARIMAGE JUSTIFY="LEFT" TILELEFT="1"

TILERIGHT="1" NORMAL=" " INACTIVE=" " HOVER=" " DOWN=" "/>

</FRAME>

<MENU/>

<CONTROL LAYOUT WIDTH="1" HEIGHT="1">

<CONTROL CLASS="Browser" KIND="A" ID="1" TOP="0" LEFT="O" WIDTH="1"

HEIGHT="1"

URL="http://www.dodots.com /dots/Calc/CALCULATOR2.htm"/>

</CONTROL LAYOUT>

</NIM>

(154) The first line of this definition establishes the identification of the NIM definition, as discussed above: it is in the domain "calculator," and the kind of display is "basic."

(155) In one embodiment, the domain will be the domain name associated with the content provider. The domain name is a unique label for the provider or developer of the NIM. The NIM's "kind" is a helpful identifier for locating the NIM, and need not be unique. A NIM may also be identified using a GlobalID, when the NIM is being shared. Since the NIM defined by this XML file is not being shared, it does not have a GlobalID.

(156) The second line of the example XML NIM definition establishes the size and appearance of the NIM frame, defining a NIM viewer in which the NIM content will be placed. The third line ensures that the height and width of the frame are fixed--that is, the size of this frame cannot be adjusted by the user. The fourth and fifth lines establish the title of the NIM--"Basic calculator"--and its location. The next four lines establish the location and placing of the titlebar and bottombar, and relevant images, e.g. mouse-over. Thus, the first part of the example NIM definition defines the NIM frame. The definition of a frame, titlebar, menu and other aspects distinguish a NIM from a browser--the content provider has control over the frame size and every aspect of the NIM's appearance, whereas when a browser is used, the content provider has to adapt to the browser display size, and browser titlebar, menu, logo and other aspects cannot be controlled by the content provider.

(157) The rest of the NIM definition identifies, positions, and initializes the NIM's controls, which, in this case, are contained by the NIM frame. In this example, the next few lines

establish that a single control will start in the upper left corner of the NIM viewer, that the control is of the type "browser," or web-rendering, and that the initialization URL for the control is www.dodots.com /dots/Calc/CALCULATOR2.htm. This URL is typically referred to as the "initialization URL," because it is where the NIM looks for NIM content when it is opened. Where the control is of type "browser," the content will typically be HTML content. However, any standard Internet content--HTML, dHTML, flash, streaming media, or Java, for example--may be used. As discussed earlier, a control, could include types other than a browser. The final two lines of the XML file close the definition file.

(158) It will be appreciated that the NIM is designed such that content consumes the entire frame. In one embodiment, the content for the corners of the frame--the menu and the exit button--and the frame sizing images are served by an application server and referenced when the user logs in. Everything else is developed and served by a separate NIM developer. This differs fundamentally from the current approach to providing web content, in which there is a strong distinction between the viewer application--the browser--and the web page or web content. Using present browser-type technology, the content is trapped within the viewer. To obtain a cohesive application feel and access to application features, the current alternative is to develop custom client applications. NIMs allow a developer to provide an application feel without developing custom client applications.

(159) NIMs and the client parser application have a messaging architecture--the NIM messaging architecture, or NMA--that enables NIMs, controls, and the client parser application to communicate. Messaging, in combination with the NIM definition, gives the content of a NIM access to the application/rendering program--the client parser application--and to other NIMs, allowing true application behavior. The NIM definition,

discussed above, is accessible, flexible, and may be changed by a NIM or a user while the NIM is in use, even after it's been rendered.

(160) The content provider, the user, or other NIMs can change a NIM. For example, the content--which includes the titlebar and menu elements--may be changed by the NIM provider by simply enhanced NIM content, using 20 messaging, so that NIMs can exhibit true application behavior. For example, an online brokerage firm can go beyond providing a NIM that renders stock tracking charts, and allow users to trade on-line via a second NIM that can interact with other NIMs such as the first NIM to help facilitate the activity.

(161) A NIM can be changed by its content, or by another NIM, using messaging. This enables a NIM to, for example, notify a user of events, such as a change in content. A NIM could, for example, remain open, but in a collapsed mode, until a particular event occurs, and could then either expand to normal size, or open another related NIM. For example, if a particular stock hits a predetermined price, the stock tracking chart NIM discussed earlier can notify the user by expanding, by popping up a message, or by opening another NIM (such as a stock trading NIM). Alternatively, the NIM could notify the user of a particular event by coming into focus or changing size or content. These changes could also be made by sending messages from a content or applications server to the NIM.

(162) As illustrated in FIG. 14, all elements of the system can send and receive NMA messages. Message routing between NIMs, from a NIM to itself (that is, between e.g. the frame elements and a control, or one control and another), and from NIMs to the system, are handled by a message interface module 106, which is part of the client parser application 38 in the client 20. In one embodiment, the message interface module 106 resolves addressing queries, executes system-level commands from the NIMs such as

"close all NIMs," and passes messages between NIMs. The message interface 106 may also communicate messages to controls such as the browser class controls, for example "navigate the addressed NIM's browser control to the argument URL." Alternatively, the message interface module 106 may route a message to an application or content server (not shown in FIG. 14) for expanded functionality. In one embodiment, the message interface module 106 uses an HTTP request to access an application program interface (API) call, with data optionally being sent and received in XML format. For example, the message interface 106 could send a message providing user input, such as credit card information to a credit processing module on a web server 82.

(163) In one embodiment, an NMA message has two components: a recipient, or address, and the message body. Both are represented as strings. The address may specify an exact NIM, a kind of NIM or control, a domain, or the system, meaning the overall home NIM display. For example, in one embodiment, the address may be in the form:

#<NTM specifier>:<control specifier>

(164) if the communication is between NIMs in the same domain, or

#<domain>:<NIM specifier>:<control specifier>

(165) or

#<domain>:<NIM specifier>:<NIM id>:<control kind>:<control id>

(166) if the communication is between NIMs in different domains, or

#system

(167) if the communication is to the system.

(168) In one embodiment, if the address is not properly specified it defaults to #system. The message interface 106 in the client 20 can flexibly allow, restrict MM addressing or sending messages--for example, the message interface can ensure that only certain NIMs can send message to a particular NIM. This allows NIM developers to develop coordinated NIMs that can interact, by sending messages to e.g. change content or open one another, without allowing other NIMs to interact with their coordinated NIMs.

(169) In one embodiment, a specifier in n address may be the unique identification of the NIM, control or domain in question: <specifier>:=<ID>. Alternatively, where the kind of NIM or control receiving the message is important but the specific NIM or control is not, the specifier may address a message to the closest matching recipient using a search criteria: <specifier>:=<kind>#<search criteria>. The kind should be a type of NIM or control that is installed in the system. Search criteria may be, for example, "any," "open," or "closed." Finally, where the control is being specified, the specifier may be a symbol, such as ".", indicating that the message is addressed to the sending NIM. For example, in one embodiment, the address #system sends the message to the system. The address #. sends the message to the NIM which sent the message. The address #7 sends the a message to the NIM with the identifier 7. The address #B#open:5 sends a message to the control with the identifier 5, in the first open dot of kind B found in the system.

(170) In one embodiment, certain NIMs will have privileges to send particular messages to certain NIMs, and not to other NIMs. For example, NIMs in a domain may be permitted to send control messages to other NIMs in the same domain, but not to NIMs in other domains. Thus, a NIM provider may have control over messaging between NIMs in his domain, and prevent NIMs in other domains from changing NIMs in his domain. Alternatively, NIM providers may coordinate with NIM providers in other domains, permitting certain messaging privileges between some of the NIMs in their respective domains. The HomeNIM and other system NIMs, such as the login NIM, which are in the system domain, may have certain messaging privileges that no other NIMs have, such as logging the user out or closing all the currently-open NIMs.

(171) In one embodiment of the invention, the second part of the message, the body, is represented as a string of characters. Messages may be sent to the client parser application, to the frame of a NIM, or to a control. Messages may be specific, defined messages, as shown in the following examples, or may be any javascript, which may be sent in or out of NIM content. Examples of defined messages to the client parser application, in one embodiment, are:

Application Message<arg>	Function
Refresh	Refreshes the user's profile.
#have-NIM <NIM-address>	Check if the user has the specified NIM as part of the user's profile.
#delete-NIM <NIM-address>	Remove the specified NIM from the user's profile.
#get-screen-width	Returns the width of the screen.
#get-screen-height	Returns the height of the screen.
#close-all-NIMs	Closes all open NIMs.
#get-NIM-ids <NIM-address>	Returns the NIM ID of the specified NIM.

(172) Messages can also be sent from a NIM to itself, or to another NIM, and the identified actions or functions, specified in the body, are performed on the receiving NIM. The following are examples of messages to a NIM for one embodiment of the invention:

Defined NIM Message<arg>	Function
#set-size<width><height>	Sets the size of the NIM.
#set-width<width>	Sets the width of the NIM.
#set-height<height>	Sets the height of the NIM.
#set-position<x-pos><y-pos>	Sets the position, of screen, of the NIM.
#set-title<title>	Sets the title of the NIM.
#collapse	Collapses, but does not close, the NIM.
#uncollapse	Uncollapses the NIM.
#set-user-sizable<width true/ false><height true/false> sizeable	Establishes whether the NIM is by the user
#set-background-color<color>	Sets background color of the NIM.
#set-title-text-justify<justify-keyword>	

(173) The above examples of NIM messages may, in one embodiment of the invention, be sent to a NIM by another NIM. For example, a user may provide input to a NIM, for example a stock tracking chart NIM, indicating an interest in another NIM, such as a stock trading NIM. The current NIM may then send an "open" message to the second NIM to open it, if it wasn't already open. The current NEM may then send a navigate message (see below) that may include an argument, such as a URL or other content pointer, so that the second NIM could be opened to a specific URL indicated by the first NIM.

(174) Finally, messages may be sent to a control of a NIM, either by the NIM itself, another control, the HomeNim system, or another NIM. As examples, the following control messages are provided in one embodiment of the invention:

Defined Control Message	Function

##<any javascript>	Run any javascript in the control, e.g. javascriptin a browser-type control.
#show	Set visibility control within a NIM.
#hide	Set invisibility of a control within a NIM.
#get-size	Get size of control.
#get-address	Query for unique ID of a control by kind.
#navigate	Navigate the control.

(175) Control messages may be used by a NIM, addressed to its own control, or to the control of another NIM. Control messages may also be sent by the HomeNIM, or by the applications server or content server.

(176) An example of a message to a NIM is window.external.PostMessage("#.:","#collapse)" this is a message from a NIM, to itself, collapsing the NIM. Another example of a message from a NIM to another NIM is "window.external.PostMessage("#mp3#any:","#open")", which is addressed to a NIM with the kind "mp3," but only if it is in the same domain as the sending NIM. The body of the message instructs the message of type mp3 to open. An example of a javascript message is "window.external.PostMessage("#mp3#any:1", "33play( )")", which sends a message to the mp3 NIM control that calls the javascript function play( ).

(177) The operation of the home NIM and individual NIMs has been described. Attention presently turns to different techniques used in accordance with the invention 25 to host NIMs.

(178) As shown and described in relation to FIG. 1, the application server 50 includes a NIM Management Modulel 12, a NIM Templates Database 74, a ShareLink Database 78, and a user profile database 76.

(179) FIG. 15 is a diagrammatic illustration of an embodiment of the NIM Management Modulel 12. NIM Management Modulel 12 may contain the NIM Templates Database 74 and the ShareLink Database 78, discussed in further detail below. NIM Management Module 112 may also contain content 402 for filling in a NIM frame or for rendering Internet pages. Alternatively, content 402 may be stored elsewhere, such as on a Web server similar to the server 82 shown in FIG. 1. As discussed above, the content preferably contains Internet content such as HTML (Hypertext Markup Language), dHTML, and images.

(180) In addition, Management Modulel 12 preferably contains executable procedures 403 for controlling and managing the NIM system. These procedures 403 may include: a Collection procedure 404 for obtaining new NIMs; a Sharing procedure 405 for sharing NIMs with others; Development procedures 406, such as a DevZone procedure 407 and a DevNIM procedure 408, for creating, modifying or deleting NIMs; Publishing procedures 409, such as a PubZone procedure 410 and a PubNIM procedure 411, for publishing NIMs so that they are publically accessible; and Administration procedures 412, such as an AdminZone procedure 413 and an AdminNIM procedure 414, for administering the system. It should be noted that the NIM sharing procedure, discussed in further detail below, may be processed by either the Server module (102 of FIG. 1), or the Sharing procedure (405 of FIG. 15). Control and management of the server and the NIM Management Modulel 12 components will now be discussed in further detail.

(181) FIG. 16 is a diagrammatic illustration of an embodiment of the NIM Templates database 74. In this embodiment, NIM Templates database 74 primarily stores XML NIM definitions in their initial unmodified state as initially designed by a partner and which have

not been altered by a user in any way. The unmodified NIMs are also referred to as "Raw NIMs". The NIM Templatesdatabase 74 is used as the starting point for the development of NIMs which may later be customized or modified by a user, developer, or system administrator, as discussed below.

(182) For ease of explanation, the individual or organization that controls the server (50 of FIG. 1) will hereinafter be referred to as the system provider and the individual or organization who supplies the content will hereinafter be referred to as a partner. It should be understood that a provider, partner, user, developer, and administrator of the system may be distinct entities, the same entity, or a combination of both. Furthermore, as discussed above, each of the above entities is assigned access rights or privileges which permit or forbid that entity from performing different actions on the system.

(183) FIG. 16 illustrates an embodiment of a NIM Template database 74. A NIM identification number (NIM_ID) 416 is stored in a NIM Template table 41.5 in the NIM Templates database 74. A Raw NW, identified by its NIM_ID, includes a plethora of RAW NIM characteristics, including, but not limited to, a Raw NIM creation date 417, which indicates when the NIM was created and is useful when searching for NIMs created during a specific time, a Raw N.I.M definition module 418, and the NIMIndex categories 422 in which the NIM has been categorized. Each NIM when created in typically classified into one or more NIM Index categories by the developer, such as "Applications", "Business", "Entertainment" and "News".

(184) Each NIM is fully configurable and definable. The NIM definition module 418 contains details defining the NIM, such as the look-and-feel 419, of the Raw NIM, initialization URLs (Uniform Resource Locators) 420, and a location 421 of where the developer would

like the NIM to open on a users's computer screen. The look and feel of the NIM is the appearance and function of the NIM interface. The look and feel may characterize the frame or skeleton layout, the graphics used to represent certain functions, such as opening and closing the NIM, whether the frame is sizable, and the appearance and operation of menus in the frame.

(185) The definition module 418 may also contain Initialization URLs 420 which reference resources containing content. The content resources may be HTML (Hypertext Markup Language), dHTML, images, programs such as Java applets, or any other file supported by HTTP. The Initialization URLs 420 contains the name of the protocol required to access the resource, a domain name that identifies a specific computer on a network, such as the Internet, and a hierarchical description of a file location on that specific computer. These files or resources are then used by the home NIM to fill in the frame and controls with content. In addition, the definition module 418 may contain other details such as the location 421 on a user's computer screen where the NIM should initially open.

(186) The NIMIndex may be used to search for, learn about, and collect NIMs. The NIMIndex is typically accessed from either a Web browser, such as Internet Explorer.RTM. or Netscape Navigator.RTM. or from the home NIM, 108 of FIG. 1. A user may search for NIMs by, or according to, any field of the NIM Templates table 415 via the NIMIndex.

(187) A user accessing the NIMIndex from a Web Brow typically navigates to a main NIMIndex web page such as that shown in FIG. 17. FIG. 17 is an illustration of the main NIMIndex Web page 423. A user typically navigates to main NIMIndex Web page 423 from a NIM home-page (not shown), or while anywhere within the NIM Web site by

clicking on the "Collect the DOtS.TM." link 424 in a menu 425. A user may search the NIMIndex by entering a search term in a form 426 and clicking on the "Search" button 428 which implements the Collection Procedure (404 of FIG. 15) to search the NIM Templates database (74 of FIG. 15) for NIMs that match the query. In one embodiment, the NLMIndex may be searched by NIM title, NIM description, or partner, as shown in the pull down menu 430. The user may also browse the NIMIndex by clicking on a link 432 to a NIMIndex category 434 which will navigate the user to a NIMIndex category Web page 440, as shown in FIG. 18.

(188) FIG. 18 is an illustration of a single NIMIndex category, the "Applications" category 435, shown in FIG. 17. A list of sub-categories (not shown), as well as a list of NIMs 442 and their short descriptions are shown in FIG. 18. A user may click on the "more detail" link 444 to be taken to a page displaying a full description (discussed later in relation to FIG. 29B) of the NIM, shown in FIG. 19.

(189) Navigation of the Internet generally occurs through the use of URLs (Uniform Resource Locators), which are the addresses of files or resources accessible on the Internet. The type of resource depends on the Internet application protocol. Using the World Wide Web's protocol, the Hypertext Transfer Protocol (HTTP), the resource can be an HTML (Hypertext Markup Language) page, an image file, a program such as a Java applet, or any other file supported by HTTP. The URL contains the name of the protocol required to access the resource, a domain name that identifies a specific computer on the Internet, and a hierarchical description of a file location on the computer and usually takes the form: "URL=protocol://machine.name[:port]/directory/document.name`?[&argu- ments] " The "protocol" is the Internet protocol used to reach the document or resource. On the Web, the

"protocol" is typically HTTP, but it can take any number of forms, such as ftp (file transfer protocol), file (a local file), gopher (gopher protocol), mailto (electronic mail address), news (Usenet news), telnet and tn3270 (interactive sessions), wais (wide area information servers), or the like.

(190) The "machine.name" is the name of the host come where the document resides (such as www.NIM.com). The ":port" portion of the address is optional and is only necessary when a resource is accessible through a non-standard TCP port number. Although the standard port number for HTTP is 80, there are numerous Web servers on the Internet that use non-standard ports, such as port 8000.

(191) The NIM system, however, may also utilize a proprietary NIM protocol. An example of a URL using the proprietary NIM protocol is: "NIMS:?NIMTemplate=<N/M_ID>"

(192) The NIM protocol URL is used to collect, distribute, and share NIMs. When collecting NIMs the NIM protocol URL is referred to as a NIMLink. When distributing or sharing NIMs the NIM protocol URL is referred to as a ShareLink.

(193) The "NIMs:" term defines the NIM protocol or scheme and is always followed by a colon. The "?NIMTemplate=<NIM_ID" is an argument, where a dollar sign (5) and a question mark (?) are used to denote path and/or search elements. It should be noted that no path is supplied (i.e.: "//path/to/something"). The argument instructs the client parser application (38 of FIG. 1) how to handle a user's selection of a NIM protocol URL and what the NIM protocol URL must do. For example, to obtain a NIM, the argument might read "NIMTemplate=123`, to obtain a Share (discussed below) the argument might read "Share=123", to obtain a Pack (discussed below) the argument might read "Pack=123", etc. The argument can be used to cause the client parser application to do anything within it's

system of functionality by specifying new argument sets to build new types of special client parser application links.

(194) In one embodiment, the address for where the client parser application (38 of FIG. 1) searches the system (10 of FIG. 1) for the NIM Template or ShareLink database (74 and 78 of FIG. 1) is specified within the processed login script or session_config, although it could alternatively be specified within the NIM protocol URL. When a user clicks on a NIM protocol URL (from any where you can place and click on a link, for example in a browser, in a NIM, in email, in a document, etc.), the client parser application processes the NIM protocol URL in the same manner as a browser processes HTTP links and an email program processes mailto links.

(195) When a protocol URL takes the form of a NIMLink, the client parser application responds by obtaining the NIM definition from the NIM Template database, optionally adds the NIM to the user's processed user profile (unless the NIM has been specified to be opened in transient mode, which may be specified in the argument), and optionally opens the NIM on the user's display screen.

(196) In one embodiment, by default, unless specified otherwise, a NIM will be added to a users collection (transient--false) and will be opened (open--true). A NIMLink with arguments may look as follows: "NIMS:?NIMTemplate&transient=true" or "NIMS:?NIMTemplate&open=false". More than one additional argument could be added by appending another argument to the URL which may read as "&argument=value".

(197) A ShareLink (discussed below) is similar to a NIMLink and may read "NIMS:?share=123", where 123 is the SHARE ID referencing the share module within the

ShareLink Database. Pack Links (discussed below) typically read as "NIMS:?pack-123", where 123 is the PACK_ID referencing a pack module within the NIM Template Database.

(198) FIG. 19 is an illustration of a full description of NIM content 446. A graphic of the opened NIM may also be displayed 448.

(199) Once the user decides that he would like to add a NIM to his home NIM, the user clicks on the "get it now" or "Get This Dots.TM. NIMLink 450 (FIGS. 18 and 19) which either runs the Collection procedure (404 of FIG. 15) which obtains that NIM's NIM definition module (418 of FIG. 16) from the NIM Template table (415 of FIG. 16), or opens another Web page as shown in FIG. 20.

(200) FIG. 20 is an illustration of a Web page 452 which might be displayed to the user once the user has clicked on the NIMLink 450. The user is presented with an option of either collecting the NIM 456, or if the user does not have the home NIM application, the user may first download the home NIM by clicking on "Get the homeDotIm" 454. Once the user clicks on the download the NIM button 456, the Collection procedure (404 of FIG. 15) obtains that NIM's NIM definition module (418 of FIG. 16) from the NIM Template table (415 of FIG. 16).

(201) The Collection procedure (404 of FIG. 15) transmits the NIM definition to the user's home NIM, which optionally opens the MM and saves the NIM definition module (418 of FIG. 16) on the user's local processed user profile. All NIM definition modules (418 of FIG. 16) on the user's computer may subsequently be saved to the user profile database, as discussed earlier in this writing. Alternatively, a "preview" button may be provided which transiently displays the NIM on the user's computer screen without adding the NIM to the

user's local processed user profile. The user may also search the NIMIndex from their home NIM.

(202) FIG. 21 is an illustration of the main home NIM graphical user interface (GUI) 464, similar to that shown in FIG. 5. The home NIM displays a list of all NIMs 466 that the user has collected. Furthermore, any NIMs that the user has collected in groups or packs, can be accessed by clicking on the "My Dotpacks'" tab 468. One way to obtain new NIMs is to click on the "Get" button 470, which opens the NIM shown in FIG. 22.

(203) FIG. 22 is an illustration of a get new NIM GUI 474. A list of all NIMs 476 (or a featured subset) that may be collected by the user are displayed. Clicking on the "More Dots'" tab 478 (shown in FIG. 23) displays further NIMs which may be collected.

(204) When a user selects or clicks on any of the NIMLinks 480, NIMLink 480 references the NIM_ID (416 of FIG. 16) for that NIM in the NIM Templates database (74 of FIG. 16). The collection procedure (404 of FIG. 15) receives the NIM_ID (416 of FIG. 16) from the user, locates the NIM definition module (418 of FIG. 16) corresponding to that NIM_ID in the NIM templates database, and transmits the NIM definition module to the user's computer. That NIM may automatically be opened on the user's computer screen. The NIM is saved to the user's list of NIMs on their home NIM (466 of FIG. 21), and the NIM definition module is saved in the user's local processed user profile. Alternatively a "preview" button may be provided which transiently displays the NIM on the user's computer screen without adding the NIM to the user's local processed user profile, as discussed above in relation to the N1MLink. All the NIM definitions that the user has listed on their home NIM are saved to the user profile database either periodically, at a set time, by event, or when the user closes their home NIM. The technique of the invention facilitates a viral distribution architecture.

In other words, the technique of the invention facilitates rampant distribution of generated NIMs, as described below.

(205) Users (or developers) may share NIMs they have collected, and perhaps even modified, with other users (or developers) in accordance with this viral distribution architecture. Because the NIM definition contains basic reference information, such as data to instantiate the NIM and URLs and other references to where the NIM content is located, a NIM is easily and quickly distributed, collected, and shared. By-packaging Internet content and applications as NIMs and referencing the NIMs by NIMLinks, the system advantageously gives Internet content viral characteristics as the NIMs can easily be distributed or shared between users.

(206) Each NIM definition contains just enough information to define and initialize the NIM's components (NIM frame, controls, etc.). For example, this information may contain data to configure the skeleton or frame that is filled in by NIM content from a developer's server. The NIM definition is therefore fairly small in size (-2K), and is therefore easily distributable as an XML file or Blob (binary large object), which is communicated using the same mechanisms (HTTP/HTTPS requests) as regular Web pages.

(207) This is especially useful where a user has collected a NIM or a group of NIMs (Packs) that he would like to send to another user. For example, a user may have an online trading NIM, calculator NIM, and stock research MM all set up in various positions on his screen, and would like to share the entire Pack with a friend who is remotely connected to the Internet.

(208) To share NIMs with others, the system utilizes the Sharelink database 78 of 25 FIG. 15 and the Sharing procedure 405 of FIG. 15. FIG. 24 is a diagrammatic illustration of the

Sharelink database 78. NIM Sharelink database 78 stores a list of all NIMs shared by users, developers, or administrators, in a share table 484. Each NIM or group of NIMs shared is assigned a Share ID 486 which points to a Share module 488. Each Share Module 488 may also include a creation date 490, multiple 30 individual MM definition modules 492, or multiple packs of NIMs that have been shared (Sharepack module 494) containing multiple NIM definition modules 496 and 498.

(209) FIG. 25 is an illustration of a Share NIM's GUI 500. All dots collected by the user (466 of FIG. 21) can be shared with other users by clicking on the "Share" button 502 shown in FIG. 21. Once the user has clicked on the "Share" button 502, the GUI 500 shown in FIG. 25 is launched. The user may then highlight any of the NIMs or packs of NIMs 504 he has collected or created and thereafter share the NIMs or packs of NIMs 504 by clicking on the "Share via email" button 506. It should be noted that other means of distributing the NIMs may be used together with, or instead of, email.

(210) When users share NIMs or NIM packs, their home NIM application generates a 10 share module, which may for example be an XML Blob containing the N1M definition or Sharepack modules shared. The shared NIM XML is then sent to, and saved in, the Sharelink database (78 of FIGS. 1 and 24). The Sharing procedure 405 of FIG. 15 then automatically generates a shared link (ShareLink) that references or points to the address of the shared XML on the Sharelink database. This ShareLink is then sent or 15 distributed (via email or posted on a Web site) to other users.

(211) If a user receives shared NIM(s) or pack(s) and has a home NIM installed on his client computer, then clicking on the Sharelink adds the NIM(s) to the user's home NIM and opens the shared NIM(s) on the user's screen. If a recipient of a shared NIM does not have

the home NIM installed on his computer, then the home NIM is downloaded and installed (with the user's cooperation), the shared MM is added to his local processed user profile, and the NIM is opened.

(212) The NIM management module (112 of FIG. 15) may also be responsible for controlling and managing the development of new NIMs via the DevZone and the DevNIM discussed below.

(213) Because NIM content is based on existing Internet content standards (HTML, DHTML, GIFs, etc.) developers can create MM content using their existing Internet content development tools and methodologies. Therefore, no special hardware or software is required to develop or serve NIM content.

(214) Furthermore, as the application server (50 of FIG. 1) hosts and delivers NIM definitions from the NIM Templates database (74 of FIG. 1) developers merely define and package the NIM content without directly authoring, hosting, or serving the XML NIM definitions. Therefore, no special hardware or software is required on the developer-side to host and serve the NIM content, other than required for their regular Internet content.

(215) Two means are provided for creating NIMs. First, a Developer Zone Web site (DevZone) and second, a set of developing NIMs (DevNIMs). Both means enable NIM developers to create, define, and modify NIM definitions, and to support the NIM development process which results in XML NIM definitions being added to the NIM Templates database and NIMLinks generated.

(216) The DevZone is a Web site where NIM developers can view a list of NIMs they have defined and/or published, add new NIMs, and categorize, view, modify, or delete their existing NIMs. The DevZone is preferably rendered in a Web browser, is hosted on the

Web server (82 of FIG. 1), and is implemented with a DevZone procedure (406 of FIG. 15). To access the DevZone, the developer may typically pass through a secure portal, such as by supplying a login identity and password.

(217) FIG. 26 is an illustration of the main DevZone Web page 510. All NIMs created by the NIM developer appear in a customized NIM list 514 that may only be accessed by that NIM developer. All NIMs created by a developer appear on the NIM list 514, unless they have been deleted by the developer or by a system administrator. The NIM list may contain the NIM name 518, the date the NIM was created 520, and an indication 516 of whether the NIM is in development or accessible by the public in the NIMIndex (i.e. "in-development" or "published").

(218) To access the NIM definition (for modification or review) the developer clicks on a "modify" or "preview" link 524 as transient (e.g. to add the NIM to their home NIM for previewing and testing). By clicking on the "modify" link, the developer is taken to the NIM modification web page, as shown in FIG. 27. Alternatively, by clicking on the "Create a Dot'" button 522, the developer is taken to a web page similar to the NIM modification web page shown in FIG. 27, where the developer may create a new NIM.

(219) FIG. 27 is a partial view of a NIM modification web page 530. To modify an existing NIM, or create a new NIM definition, a developer preferably utilizes Web forms, such as 534 to 542, or any area that contains objects that capture user input, such as text entry spaces, check boxes, and selection buttons. Developers typically fill in forms with information which defines the NIM, where the details might include the NIM's name 534, the URL for any image associated with NIM 540 (as shown in FIG. 18), the URL for a detailed image 542 (448 of FIG. 19), and such details as NIM frame (e.g., size of NIM,

sizeable), layout of the controls (e.g., WebConduit control), and to specify the initial MM content (e.g., the initial target URLs for the WebConduit control, TitleBar, BottomBar), and any categories in which the developer would like the NIM to be listed in the NIMIndex. Once the developer has completed or modified the forms, he may either save or delete the NIM 532. If the developer selects either the development check box 536 or the public check box 538, and then saves the NIM, the DevZone procedure (407 of FIG. 15) generates a XML NIM definition, stores the XML NIM definition in the NIM Templates database (74 of FIGS. 1 and 15) and returns a NIMLink pointing to that NIM which is listed on the NIM list (514 of FIG. 26) on the developer's home NIM. The only difference being that once the developer selects the public check box 538 and saves the NIM, the NIM definition is published utilizing the PubZone publishing procedure (410 of FIG. 15) to a publically accessible portion of the NIM Template Table (415 of FIG. 16), from where users can access, download, and collect the NIM. If the developer selects the development check box 536, the NIM can only be viewed and or modified by the developer and system administrator. It should be noted that the DevZone only allows control of certain characteristics of each NIM. Other characteristics may be set to default while still other characteristics can only be altered by an administrator. In an alternative embodiment, the DevZone may be rendered in a NIM or group of NIMs just as it was rendered in a Web browser. In either embodiment, NIM developers fill out one or more forms specifying NIM definition parameters, an XML NIM definition gets created and stored in the NIM Templates. Database, and a NIMLink gets generated that points to the new NIM. The Developer can then view or debug this NIM by clicking on the NIMLink to add it to his home NIM, or preview as transient, and thereafter render it on his screen. NIM definitions

may also be developed using NIMs and NMA messages. A 30 developer may create Raw NIMs from empty NIM Templates using a development NIM (the DevNIM) on the developer's home NIM.

(220) FIG. 28A to 28D are GUIs of a development NIM (the Dev NIM). A developer may obtain a DevNIM by either collecting the DevNIM in the usual manner, as discussed above, or the system, via the system administrator, may share the NIM with the developer, also as discussed above. The DevNim contains a DevNIM procedure (408 of FIG. 15) which is transmitted to the developer's home NIM, as discussed above.

(221) To create a new NIM, the developer launches the DevNIM and enters a NIM name 550 into the DevNIM. The server then obtains an empty NIM (a NIM with default or no initialization data and with only basic characteristics) from the NIM Template Database using the procedure for collecting NIMs described above, and saves the empty NIM under the supplied new NIM name 550 locally in the developer's processed user profile. The developer may then modify the empty NIM to the required form using the DevNIM. In the preferred embodiment a pull down menu 552 is provided where the developer can select which feature to modify, such as the frame characteristics (FIG. 28B), the titlebar (FIG. 28C), or initialization URLs for different frame or control elements (FIG. 28D).

(222) Each time the developer modifies a setting, the DevNIM, using the DevNIM procedure, sends NMA messages to the newly saved NIM to modify its definition parameters. For example, modifying the NIM's name, size, TitleBars, BottomBars, or WebConduits (as shown in FIGS. 28A to 28C).

(223) Unlike the DevZone, a new XML NIM definition and NIMLink is not generated every time a modification is made. All modifications (during the development cycle) are made

locally to the NIM definition and are stored in the developer's processed user profile. The DevNIM embodiment, therefore, requires a separate publishing step that promotes the newly created NIM definition from the developer's user profile, to the NIM templates database on the application server.

(224) To publish the NIM, the NIM developer categorizes the NIM and the NIM definition is copied from the developer's processed user profile to the NIM Templates database.

(225) A publishing NIM (PubNIM), implemented with Publishing procedures 409, is provided to handle these functions. The PubNIM may therefore be shared or transmitted to the developer along with the DevNIM. The PubNIM contains a PubNIM procedure (411 of FIG. 15) which controls the publication of the NIMs to the NIM template database, as discussed above. The PubNIM procedure sends a NIM definition module to the application server which receives the NIM definition module, extracts the NIM definition from the share module, stores it in the NIM Templates database, and associates the NIM with the developer so that the NIMLink shows up on the developer's NIM list (in their DevZone account).

(226) Alternatively, the new NIM may be published directly from the DevNIM. Once a user is satisfied with the NIM, he may select an option which publishes that NIM definition to the NIM Templates database. It should be noted that a developer may modify his NIMs at any time from the DevNIM.

(227) As mentioned earlier, when a developer is first authorized to create and/or modify NIMs, or at any time thereafter, information about that developer is saved in that developer's user profile (76 of FIG. 1) on the application server.

(228) A developer may also create application programs using NIMs, which a user may access from his client computer. Just as client-side application characteristics (sizing, position, menus) are accessible to content via NMA, the system may offer server-side application functionality, or toolkits, which are accessible through the NMA.

(229) A developer can build a NIM application without implementing, hosting, or supporting complex server or client applications. By using the server toolkits, a developer can develop N1Ms that exhibit server-application behavior by focusing on implementing NIM content (just like standard Internet content).

(230) For example, a NIM's content (an HTML page) may send a message to the system (or server) to request a credit card to be processed. Other toolkit examples may include credit card billing, user profiling, targeted advertising, email, chat rooms, Internet telephony applications, or calendars.

(231) Any server-side application can be made accessible through the NMA, as a toolkit, just as client-side application behaviors are made accessible. In the current implementation, server-side application functions could be offered by a NIM (exposed via javascript functions on a page in a hidden frame). Other NIMs could access this functionality by sending NMA messages to this "Toolkit NIM" calling the functions. A NIM developer may therefore focus on Internet content development while accessing the features, behaviors, and functionality of an application just as if he had developed custom client and server side applications. The NIM management module (112 of FIG. 15) may also be responsible for controlling and managing the administration of the system via the AdminZone and the AdminNIM discussed below.

(232) A system administrator has the power to create, modify or delete users, developers, NIMs, other administrators, or NIMIndex categories, depending on that administrators access privileges. In a similar manner to the DevZone and DevNIM, system administrators may utilize either a Web browser administration zone (AdminZone), or an administration NIM (AdminNIM) which both make use of Administration procedures (412 of FIG. 15).

(233) To access the AdminZone, an administrator typically passes through a secure portal, such as by supplying a login identity and password. Once within the AdminZone, the administrator may search for a NIM by NIM name or title 552, category, developer, developer contact name, or status, as shown in FIG. 29A. The administrator may also selectively search for NIM's in development or publically accessible NIMs 554.

(234) Utilizing an AdminZone procedure (413 of FIG. 15), once the required NIM 20 is located the administrator may modify or delete the NIM in a similar manner to a developer as shown in FIG. 29B, and described above.

(235) Also utilizing the AdminZone procedure, the administrator may manage NIMIndex categories by creating new categories, modifying or deleting existing categories, and/or adjusting the layout of the NIMs within those categories as shown in FIGS. 30A and 30B. For example, an administrator may change a category's name 558, designate the category active or inactive 556, or create sub-categories 560. The system administrator may also select a category or categories for the NIM to appear in, where each NIM may be registered in more than one category.

(236) Finally, utilizing the AdminZone procedure (413 of FIG. 15), an administrator may search for users, providers, or developers and adjust their details, as shown in FIGS. 31A and 31 B. The system administrator may, for example, change a users contact details. In

addition to adding, modifying or deleting NIMs, system administrator may have the task of reviewing NIM submissions from developers and promoting NIMs to the public. A submissions list of newly submitted NIMs may be displayed to an administrator, who may promote the NIM to the public or view the NIM. Once promoted, changes are made to the NIM Templates database and the NIM is automatically removed from the submissions list (again by utilizing the AdminZone procedure (413 of FIG. 15)).

(237) The foregoing discussion has explored the inherent nature of NIMs. Attention now turns to different techniques that may be used to exploit information that is associated with the use of NIMs. In particular, the following discussion is directed toward the accumulation of statistical information that is only available in view of the architecture of the present invention.

(238) Currently, the predominant method of tracking and collecting user online behavior is severely limited for a number of reasons. First, most Internet use or visitor statistics are single-dimensional (linear, sequential) because Internet content is presented to users one full-screen page at a time. Second, users visit and leave sites so rapidly their visits are barely meaningful. Third, user's browsing habits are often discontinuous (browsers give users navigational bypass controls--back, forward, home, refresh, stop, etc.). Fourth, user behavior tracking is limited from a single site's server point of view. Current use statistics are plagued with the challenge of tracking continuous user behavior (especially from a cross-company perspective), with more than a single dimension of use context. Finally, because a computer may have multiple users, or a single user may use multiple computers, tracking continuous user on-line behavior is extremely difficult.

(239) One of the advantages of the NIM system as illustrated in FIG. 1, is that the Server 50 is able to track continuous, long-term NIM use information about each user. This is because the NIM server, through communication with the home NIM, can track each NIM event performed by each user. Therefore, it is possible to track each individual user's entire NIM use activity from the moment the user downloads the home NIM.

(240) Referring to FIG. 32, in one embodiment of the invention the following events may be tracked by the Event Log Module 98 (within the client computer 20 of FIG. 1): home NIM Download Event 704 NIM Download Event 706 NIM Display Event 708 Web Click-Through Event 710 Page-View Event 712 First NIM Installation Event 714 First home NIM Startup Event 716 Transient Mode Event 718 Share NIM Received Event 720 NIM Pack Received Event 722.

(241) A home NIM Download Event 704 is logged when the user clicks on a link to request the home NIM user application. Preferably, the start time 752, and the end time 754 are recorded for this event. Also recorded is the provider ID 750 which is a parameter (generally, an integer) that represents the content provider partner who provided the link to the user.

(242) NIM Download Event 706 is logged when the home NIM acquires a NIM via a NIMLink. The start time 752, the end time 754, and the provider ID 750 are recorded for this event. Also recorded is the NIM ID 756 which is a parameter (generally, an integer) that represents the NIM that was just downloaded.

(243) NIM Display Event 708 is logged when a user activates a NIM. The NIM ID 756, the start time 752, and the end time 754 are recorded for this event. Web Click-Through Event 710 is logged whenever a user links from a NIM to a full-screen browser. This can occur

when a user clicks on a link in the NIM, or it can occur automatically through the NIM messaging, or directly through the content provider. The NEM ID 756, the start time 752, and the Internet address 758 of the link are recorded. Page-View Event 712 is logged whenever a user views a page of content within a NIM. The NIM ID 756, and the start time 752 are recorded for this event.

(244) First NIM Installation Event 714 is logged the first, a NIM or NIM Pack is installed from a web site. This event is logged only once for each user account. The NIM ID 756, start time 752, and end time 754 are recorded for this event.

(245) First home NIM Startup Event 716 is logged when the home NIM runs for the first time. This event is logged only once for each user account. The start time 752 is recorded for this event.

(246) Transient Mode Event 718 is logged when the home NIM runs in transient mode. Transient mode occurs when the home NIM runs before the user has logged in. The start time 752, and the end time 754 are recorded for this event.

(247) Shared NIM Received Event 720 is logged for each NIM a user receives as part of a share. If a NIM Pack is shared, this event will be recorded for each NIM in the shared pack. The NIM ID 756, the start time 752, and the end time 754 are recorded for this event.

(248) Shared NIM Pack Received Event 722 is logged for each NIM Pack a user receives as part of a share. Thus, when a NIM Pack is shared, an Event 720 will be logged for each NIM in the NIM Pack, while an Event 722 will be logged once for the NIM Pack itself. The start time 752, and the end time 754 are recorded for this event.

(249) The events listed above are tracked in one particular embodiment. Other embodiments may track more or perhaps fewer events. This comprehensive event tracking is possible

because each user event can be identified by the NIM Server through communication with the home NIM. Additional events may include tracking when a user sends a share or tracking when a user sends a NIM or a NIM Pack.

(250) FIG. 33 shows a typical series of user actions 800 as they are tracked by the Event Log Module 98. First, a user may request to download the home NIM application (step 802) from either a partner's web site or the NIM Server 50. The Event Log Module 98 records a home NIM Download Event, as shown with field 704 of FIG. 32. The start time 752, and the end time 754 are preferably recorded. Also, the provider ID 750 of the site from where the home NIM download request was received is recorded.

(251) Returning to FIG. 33, the user subsequently activates the home NIM for the first time (step 804). The Event Log Module 98 records a First home NIM Startup Event 716, as shown in FIG. 32. The start time 752 is preferably recorded. In addition, the home NIM is activated and the user lot yet logged in, a Transient Mode Event 718 is logged and the start time 752 is recorded.

(252) As shown in step 806 of FIG. 33, the user logs into the home NIM. When this occurs, the end time 754 may be recorded for the Transient Mode Event 718.

(253) A user download of a new NIM (step 808) may be from a partner's web site or the NIM Server. When this occurs, the raw NIM definition is copied into the user's User Profile 76. The event log 98 records two events. First, because this is the first NIM the user has installed, a First NIM Installation Event 714 is recorded. The start time 752, the end time 754, and the provider ID 750 of the download site are preferably recorded. The second event recorded is a NIM Download Event 706. The Event Log Module 98 preferably tracks the NIM ID 756, the provider ID 750, the start time 752, and the end time 754 for this event.

The next thing a user may do is open the NIM (step 810). This consists of retrieving the NIM definition from the user's User Profile and getting NIM content from the provider 82, as discussed above. The NIM is displayed for the user and the Event Log Module 98 records a NIM Display Event 708. However, at this point, the Event Log Module 98 can only record the start time 752, and the NIM ID 756 for this event. The end time 754 is recorded when the NIM is closed.

(254) For every page of content a user views within a NIM 812, a Page-View Event 712 is recorded. Some page views may require content from the provider 82. The NIM ID 756, and the start time 752 are recorded for this event.

(255) The NIM may also enable the user to click on a link that results in navigating to a full screen web browser (step 814). When a user does this, a Web Click-Through Event 710 is recorded. The Event Log Module 98 records the NIM ID 756, the start time 752, and the URL of the web site that is passed from the NIM content to the browser 758.

(256) When the NIM closes (step 816), the end time 754 for the NIM Display Event 708 is recorded. When the user logs out of the home NIM (step 818), the event log is uploaded to the Server 50 (of FIG. 1).

(257) In one embodiment of the invention, the previously described Event Log Module 98 (within the client computer 20 of FIG. 1) tracks user events in the home NIM user application and uploads the information to the Statistics Database 80 (of the server computer 50 of FIG. 1) at predetermined intervals alternate embodiments, the Event Log 700 (in FIG. 32) may be processed by the NIM Server before it is stored in the Statistics Database 80. For example, the NIM Server may process NIM use status information for each user that is currently logged in.

(258) The Statistics Database 80, illustrated in FIG. 34, preferably lists every event 1002 by every user of home NIMs along with the corresponding fields associated with each event. For-example, if a NIM Display Event is recorded, the User ID 1004 of the user that performed the event is listed, the start time 1006 is listed, the end time 1008 is listed, and the NIM ID 1010 is listed. If a Web Click-Through Event is recorded, the User ID 1004 is listed, the NIM ID 1010 is listed, the start time 1006 is listed, and the URL of the web site 1014 is listed. The Statistics Database 80 therefore allows the list of events to be easily referenced and searched by each event or by each of the fields associated with the events.

(259) Referring to FIG. 35, the Statistical Analysis Module 900 uses the Statistics Database 80 in order to provide various services for the content provider partners 82. Preferably, the Statistical Analysis Module 900 includes a Multi-Dimensional Consumer Profile Module 902, a Real-Time Advertising Module 904, and a Pack Building Module 906, as discussed below.

(260) A primary advantage of the present invention is that, because NIMs are used in groups and are used more often and for longer periods of time than web pages or web sites, real-time multi-dimensional NIM use data (that's a function of which NIMs are activated simultaneously) can be accumulated. In accordance with an embodiment of the invention, this accumulated data is used to generate a multi-dimensional consumer profiling database. The Multi-Dimensional Consumer Profile Module 902 uses information from the Statistics Database 80 to examine, for each user, the start time, and the end time of each NIM Display Event It then determines the NIMs (using the NIM IDs) that are opened simultaneously for each user. The Module 902 determines, for every selected NIM, the other NIMs that a given user may use in conjunction with the selected NIM. The Module 902 also determines how

often these other NIMs are used simultaneously with the selected NIM. For example, Company X provides a NIM for selling its books. The Multi-Dimensional Consumer Profile Module 902 determines for Company X that a particular user has a NIM related to finance activated 30% of the time the user has the book-selling NIM acted, a NIM related to computers 20% of the time the user has the book-selling NIM activated, and a NIM related to wedding gifts 5% of the time the user has the book-selling NIM activated. This will provide Company X with a more complete profile of the user's interests.

(261) The Real-Time Advertising Module 904 determines the NIMs that each user has displayed at any given moment. This information is used by a content provider partner or by the NIM Server to target advertising information. For example, if a user has a NIM related to sports displayed simultaneously with Company X's book-selling NIM, Company X uses this information to stream an advertisement for a sports book. In one embodiment, this is accomplished by associating each NIM with a context keyword. This is done by incorporating the context keyword into the NIM definition or, alternatively, by maintaining a table of NIMs and their corresponding context keywords. For example, the NIM related to sports is associated with the context keyword "sports." Moreover, the Real-Time Advertising Module 904 may combine the real-time user information with the historical user information from the Statistics Database 80 to provide advertisers with a complete picture of a user's interests.

(262) The Pack Building Module 906 uses the Statistics Database 80 to determine which NIMs are being used simultaneously. The Module 906 also determines which NIMs are being shared as NIM Packs. From this, the Module 906 provides information to content provider partners about which NIMs should be bundled together. In alternate embodiments,

the Module 906 builds a NIM Pack based upon the information it processes. For example, if the Pack Building Module 906 determines that an airline NIM is being used with a hotel NIM and a car rental NIM, the Module 906 may build a NIM Pack with a restaurant NIM.

(263) Additionally, in one embodiment of the present invention, the NIM Server 82 may track the content within a NIM in a Content Database 1050, as illustrated in FIG. 36. A content descriptor 1052 which may be a string describing the content that is shown within the NIM is recorded for content shown in the NIM. For example, if a NIM displayed an advertisement for an automobile followed by an advertisement for a restaurant, the two recorded content descriptors might say "automobile ad" and "restaurant ad." In addition, the NIM ID 1054, the start time at which the content is displayed 1056, and the end time 1058 are all preferably recorded for each content descriptor.

(264) Referring to FIG. 35, The Content Analysis Module 950 is able to correlate, at any moment, the content displayed to the user as recorded in the Content Database with the user's NIM activity recorded in the Statistics Database. For example, if one NIM displays to a user an advertisement for a travel book, the user may open a NIM related to Florida, a NIM owned by a specific airline, and a NIM owned by a car rental company. This pattern of user behavior will allow the company that provides the travel book advertisement to better understand the effect of the advertisement on the user. The company may use this information to make cross-promotions with other NIM providers, or, simply to provide more effective targeted advertisements.

(265) In an alternative embodiment, each of the content providers may track its own content information. The content providers could then compare its content information with the user information provided by the Statistics Database of the NIMServer.

(266) Finally, referring to FIG. 37, all of the user event information may be used in conjunction with user information provided at login. During the login process, the user may be required to enter demographic information such as age, marital status, etc. In one embodiment, this information is stored in a User Account Database 1100. Each User ID 1102 is listed along with the corresponding user information 1104. Therefore, it is possible to match the user events with personal information about the particular user to give advertisers or NIM content providers a more complete behavior profile of each user.

(267) The foregoing description, for purposes of explanation, used specific nomenclature to provide a thorough understanding of the invention. However, it will be apparent to one skilled in the art that the specific details are not required in order to practice the invention. In other instances, well known circuits and devices are shown in block diagram form in order to avoid unnecessary distraction from the underlying invention. Thus, the foregoing descriptions of specific embodiments of the present invention are presented for purposes of illustration and description. They are not intended to be exhaustive or to limit the invention to the precise forms disclosed, obviously many modifications and variations are possible in view of the above teachings. The embodiments were chosen and described in order to best explain the principles of the invention and its practical applications, to thereby enable others skilled in the art to best utilize the invention and various embodiments with various modifications as are suited to the particular use contemplated. It is intended that the scope of the invention be defined by the following claims and their equivalents.

SUBSTITUTE SPECIFICATION (MARKED UP)

DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] The present application is a continuation of and incorporates by reference U.S. Non-Provisional patent application Ser. No. 09/558,925, filed Apr. 26, 2000, now U.S. Patent No. 7,660,868, which claims priority from and incorporates by reference U.S. Provisional Application ser. nos. 60/131,083, filed Apr. 26, 1999, 60/131,114, filed Apr. 26, 1999, 60/131,115, filed Apr. 26, 1999, 60/176,687, filed Jan. 18, 2000, and 60/176,699, filed Jan. 18, 2000. The present application claims priority to U.S. Non-Provisional patent application ser. no. 09/558,925, filed Apr. 26, 2000, now U.S. Patent No. 7,660,868, and each of the aforementioned applications to which it claims priority.

[0002] The present application is also related to and incorporates by reference the following U.S. patent applications: Non-Provisional Application ser. no. 09/558,922, filed April 26, 2000, now U.S. Patent No. 7,756,967; Non-Provisional Application ser. no. 09/558,923, filed April 26, 2000; Non-Provisional Application ser. no. 09/558,924, filed April 26, 2000, now U.S. Patent No. 7,356,569; Non-Provisional Application Ser. No. xx/xxx,xxx, filed October 31, 2007 titled "Component For Accessing And Displaying Internet Content" (attorney docket no. IMS 07-05-01); Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,286, filed Oct. 31, 2007, titled "Server Including Components For Accessing And Displaying Internet Content And For Providing Same To A Client" ~~(attorney docket no. IMS~~

~~07-05-02)~~; Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,392, filed Oct. 31, 2007, titled "Method For Accessing And Displaying Internet Content" ~~(attorney docket no. IMS 07-05-03)~~; Non-Provisional application Ser. No. ~~xx/xxx,xxx~~ 11/932,427, filed Oct. 31, 2007, titled "Component For Coordinating The Accessing And Rendering Of An Application Media Package" ~~(attorney docket no. IMS 07-05-04)~~; Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,456, filed Oct. 31, 2007, titled "Tracking and Tracing User Activity with Application Media Packages" ~~(attorney docket no. IMS 07-05-05)~~; Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,585, filed Oct. 31, 2007, titled "System and Methods for Creating and Authoring Internet Content using Application Media Packages" ~~(attorney docket no. IMS 07-05-07)~~; Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,630, filed Oct. 31, 2007, titled "Methods of Obtaining Application Media Packages" ~~(attorney docket no. IMS 07-05-08)~~; Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,663, filed Oct. 31, 2007, titled "Indexing, Sorting, and Categorizing Dots" ~~(attorney docket no. IMS 07-05-09)~~; Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,692, filed Oct. 31, 2007, titled "System and Methods of Messaging between Application Media Packages" ~~(attorney docket no. IMS 07-05-10)~~; and, Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,763, filed Oct. 31, 2007, titled "Component For Accessing And Displaying Internet Content In Association With a Web Browser Application" ~~(attorney docket no. IMS 07-05-11)~~.

~~**COPYRIGHT NOTICE**~~

~~**[0003]** A portion of the disclosure of this patent document contains material which is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure as it appears in the~~

Patent and Trademark Office patent file or records, but otherwise reserves all copyright rights whatsoever.

BACKGROUND OF THE INVENTION

1. Field of the Invention.

[0004] The present invention is related to accessing and viewing Internet content, and more specifically to a method and apparatus for providing a unique frame, independent of a Web Browser application and window, for the retrieval and display of such content.

2. Description of the Prior Art.

[0005] A user operating a client computer typically accesses the Internet by using a viewer application, such as a browser to view Internet content provided at a destination address, typically a web page. In this context, Internet content and web applications are designed to fill the entire web page. It is known to divide the Internet content into different regions of a single web page. For example, personalized web pages can be specified, such that a user views a variety of content sources in a single page, such as stock information, weather information, and sports information, which is aggregated at the server that delivers the web page to the user, who then views the aggregated content in a single web page. Observe that even when disparate content is aggregated, in this manner, it is reassembled into a full web page and is served through a full-screen browser.

~~**[0006]** Users and application developers therefore have limited control over the presentation of internet content: content is typically trapped within the frame of the browser. A developer's only alternative to engaging a user page-by-page in a browser is to develop, distribute, and support custom client software. In the Web browser scenario, it is the content provider, not the user that aggregates the information that is viewed by the user. Thus, the user is not in a position to separately aggregate the content at a client computer, instead the user is constrained to view the content that has been delivered in the manner provided by the server computer hosting the web page. There is a growing desire for individual users to fully control the aggregation and presentation of content and web applications that appears on a client computer.~~

~~**[0007]** A user who wishes to view multiple web pages or applications can open multiple instances of a browser. However, the user will not be able to view each "full-screen" page at the same time. Instead, the user must adjust the windows corresponding to each browser instance and view only part of each page. The information appearing in each browser is not designed for viewing in this manner. Thus, the user cannot create an optimized display of content from multiple sources.~~

~~**[0008]** Currently, content providers and end users have limited tools to alter the browser in which content appears. That is, the controls associated with a browser are not fully configurable. Thus, the vendor of a browser is in a position to brand the browser and regulate the controls associated with the browser. There is a growing desire for content providers to not only fill a browser with their content, but to also fully brand and control the frame in which the content appears. Further, in some instances, content providers desire to limit the controls~~

associated with a browser or viewer, so that a user is more inclined to view a single set of content, for example, by having limited access to previously viewed content. However, the current browser applications provide very limited control to a user or content provide to alter the frame and controls provided by the browser window.

[0009] In summary, therefore, the current model of the Internet has the following attributes and limitations:

a) Internet content is typically viewed one page at a time, with each page displayed serially displayed in a browser application window which typically takes up the majority of the user's computer screen real estate.

b) Internet content is designed for delivery in web pages. Even if content is modularized, it is reassembled into a full web page and viewed serially in the window of browser application taking up a significant portion of the user's desktop.

c) There is a distinction both visually and architecturally between the "viewer application" (browser) and the "content/document" (web page) such that a browser window is not tailored to the content being displayed, but rather is capable of displaying any web content.

d) Internet content is effectively limited and trapped within the "frame" of the browser (viewer application). Therefore, content developers, users

and web-application developers are limited in how the user experience is controlled.

e) Although internet programming technologies (such as Java script, CSS, layers, flash, etc.) are giving web pages more functionality, the pages have limited access to application functionality such as access and control of the window and frame, the size of the frame, branding, application behavior such as size and menu items, etc.

f) "Web-applications" such as web-mail and web calendars are being packaged and viewed through the page-by-page web model. Even though web-applications are being implemented by many online companies, the web is currently a destination page-based model where, for example, a user visits one page, then goes to another page and so on. It is therefore a sequential, linear experience, e.g., one full page at a time.

[0010] In view of the foregoing, there is a need in the art to provide a technique for accessing multiple instances of distributable computer readable web content in which these instances are typically smaller than the full pages used in current web pages and web applications, and which may be displayed in user- or content provider-controlled frames. Preferably, such techniques allow such access to be done simultaneously. There is a further need for providing the user with flexibility in selecting, collecting, relating and viewing such web content and for giving the content provider flexibility in directing media to a specific user and controlling the framework in which media, such as

web content, is presented. Finally, there is a need to gather more accurate information regarding the type of content that a user enjoys, so that the user can be automatically provided with this content.

SUMMARY OF THE INVENTION

[0011] Accordingly, the present invention is directed to systems and methods by which predefined web content or predefined references to web content may be accessed and displayed on a web-connected client computer. The method relies on the structure defined herein as an Application Media Package. Application Media Packages are web browser-readable code that is executed on a non-browser-based installed client application. The client application, referred to herein as an Application Media Viewer, executes independently from a web browser. The Application Media Viewer parses and executes the Application Media Package code to create the user experience. The Application Media Viewer is capable of loading and simultaneously running a plurality of different Application Media Packages.

[0012] Application Media Packages are programmed to access and display web data including media content. By design, Application Media Packages may graphically reflect the type of data as well as time varying changes in the data. Unlike web pages which have formatting restrictions that are dictated by the browsers into which they are loaded, Application Media Packages have a graphically definable format which is the graphical user interface (GUI) as presented on a user's client computer. Furthermore, unlike web pages together with

~~browsers, Application Media Packages are not generally navigable. They are typically constructed to represent specific data for a specific application.~~

~~**[0013]** Each Application Media Package contains both code and presentation display data or network reference to that data. Together with the Application Media Viewer, the program combination is capable of accessing remote server data that may be processed and/or displayed, modifying on-screen appearance or functionality according to user events, processing web events, client computer events, etc.~~

~~**[0014]** The Application Media Package may operate in a windowed operating system and uniquely operate without the use of a generalized application window. Therefore the Application Media Package graphical user interface is wholly definable by the Application Media Package software developer. It may therefore be designed to exclude the window display requirements associated with the conventional window based operating system applications.~~

~~**[0015]** Application Media Packages may be coded independently of the Application Media Viewer client platform. Thus, independent third party developers may create and post Application Media Packages according to guidelines provided by the creator of the Application Media Viewer. As Web users are increasingly reluctant to download applications from the Web and onto their computers, the Application Media Viewer provides a single trusted application that is downloaded and installed one time.~~

~~**[0016]** The Application Media Viewer then provides a layer of security for Web users so that they may download Application Media Packages without the same exposure to compromising their computer as with installed applications. This security is provided by virtue of a trusted module platform together with the module's web browser readable code. This is in contrast to a typical client application which is directly machine executable and may have broader access privileges to the client computer's resources.~~

~~**[0017]** According to an aspect of the invention a physical memory device is provided having stored thereon a first computer software component comprising: a definition of a graphical user interface within which content in a web browser readable language may be presented; an element for causing said user interface to be rendered outside of and without utilization of a web browser application; an element for causing said content to be presented in said graphical user interface; and a network reference to said content for presentation within said user interface.~~

~~**[0018]** The above is a summary of a number of the unique aspects, features, and advantages of the present invention. However, this summary is not exhaustive. Thus, these and other aspects, features, and advantages of the present invention will become more apparent from the following detailed description and the appended drawings, when considered in light of the claims provided herein.~~

~~**BRIEF DESCRIPTION OF THE DRAWINGS**~~

~~**[0019]** In the drawings appended hereto like reference numerals denote like elements between the various drawings. While illustrative, the drawings are not drawn to scale. In the drawings:~~

~~**[0020]** Fig. 1 is an illustration of the overall architecture of a network in which the present invention may operate.~~

~~**[0021]** Fig. 2 is an illustration of the elements of an Application Media Package (Dot) according to one embodiment of the present invention.~~

~~**[0022]** Fig. 3 is an illustration of the elements of an Application Media Package (Dot) definition according to one embodiment of the present invention.~~

~~**[0023]** Fig. 4 is an illustration of an instantiation of a generic GUI according to one embodiment of the present invention.~~

~~**[0024]** Fig. 5 is an example of an instance of an Application Media Package (Dot) according to one embodiment of the present invention.~~

~~**[0025]** Fig. 6 is an illustration of the communication flow associated with the acquisition and instantiation of an Application Media Package (Dot) according to one embodiment of the present invention.~~

~~**[0026]** Fig. 7 is all illustration of the communication pathways associated with the building of a user profile, and the acquisition of Dots through links, packs, and sharelinks according to one embodiment of the present invention.~~

~~**[0027]** Fig. 8 is an illustration of the communication pathways associated with the acquisition and instantiation of an Application Media Package (Dot) according to one embodiment of the present invention.~~

~~**[0028]** Fig. 9 is an example of a display showing Application Media Packages (Dot) available for downloading as well as Application Media Packages (Dots) already downloaded according to one embodiment of the present invention.~~

~~**[0029]** Fig. 10 is an illustration of the communication pathways associated with the instantiation of an already-acquired Application Media Package (Dot) according to one embodiment of the present invention.~~

~~**[0030]** Fig. 11 is a flow chart illustrating the download process of an Application Media Package (Dot) and Application Media Viewer (Home Dot) according to an embodiment of the present invention.~~

~~**[0031]** Figs 12A, 12B, and 12C are illustrations of Application Media Packages arranged in groups, and arranged in vertical and horizontal Blocks (position-justified groups), respectively, according to an embodiment of the present invention.~~

[0032] Fig. 13 is an illustration of the client-server model of a system according to the present invention.

[0033] Fig. 14 is an illustration of the message routing paths and elements of a Dot Messaging Architecture according to the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0034] The present invention discloses a technology that is capable of processing distributable computer readable media. Distributable computer readable media includes, but is not limited to, standard Internet content, such as HTML, dHTML, images, imbedded ActiveX and Java applications, JavaScript, CSS, Perl scripts, Streaming Media, and/or Flash. The present invention is advantageous relative to prior art systems and methods because it provides improved mechanisms for simultaneously interacting with several independent sources of distributable computer readable media, collecting references to such media, and sharing such references with other users. The disclosed technology is further advantageous because it provides improved systems and methods for on-screen management of distributable computer readable media.

[0035] Central to the present invention is the concept of an Application Media Package. As used herein, the term Application Media Package refers to a component which includes a fully configurable frame with one or more controls; the frame through which content is optionally presented. The fully configurable frame utilized in

accordance with the invention stands in contrast to present web browsers, which are branded by the browser vendor and which have limited means by which to alter the controls associated with the browser.

ABSENCE OF WEB BROWSER

[0036] The Application Media Package is a file that is comprised of web browser readable language. According to the preferred embodiment, the present invention renders, displays, and updates Internet data without the use of a browser. In fact, no browser need be installed on the client computer on which the Application Package is instantiated. The present invention produces a user experience by parsing and rendering the Application Media Package through the Application Media Viewer. The Application Media Viewer is an installed client application which renders the Application Media Package as well as provides additional functionality to the user experience (hence, it is more than simply a viewer application). The Application Media Viewer may render web browser readable content (of the type typically supported by a browser application) due to it being programmed utilizing the Microsoft 1E4 object for receiving, parsing and rendering web content. It will be understood by one skilled in the art that, despite its name, the Microsoft 1E4 object is not a web browser application. One apparent difference between the Application Media Viewer and a typical web browser application is that the Application Media Viewer of the present invention is not designed to provide user web navigation and page control typically provided by a web browser.

~~**[0037]** Fig. 1 is a general illustration of a system in accordance with one embodiment of the present invention. In Fig. 1, a network 10 is operated in accordance with the present invention. Network 10 includes at least one user or client computer 20, at least one server computer of class 50, and optionally one or more server computers of class 82. User computer 20 as well as server computers of class 50 and 82 are each connected by transmission channel 44, which is any wired or wireless transmission channel.~~

~~**[0038]** User computer 20 is any device that includes a Central Processing Unit (CPU) 24 connected to a random access memory 30, a network connection 28, and one or more user input/output ("i/o") devices 40 including output means 42. Thus, the term "computer," as used herein, is intended in its broadest sense to include not only traditional workstations, laptops and similar devices, but any device containing a CPU of sufficient operability to utilize Internet data, such as web-enable cell phones, personal digital assistants (PDAs), and so forth.~~

~~**[0039]** Output means 42 is any device capable of communicating with a user and includes, for example, a video monitor, a liquid crystal display, voice user interfaces, and/or integrated graphic means such as mini-displays present in web-enabled cell phones, PDAs, etc.~~

~~**[0040]** Typically, user computer 20 includes a main non-volatile storage unit 22, preferably a hard disk drive, for storing software and data. Further, user computer 20 includes one or more internal buses 26 for interconnecting the aforementioned elements. In a typical embodiment, memory 30 includes an operating system 32 for managing files and programs associated with user computer 20. In some embodiments, operating system 32 includes a registry 34 that has one or more references to specified locations in system 10.~~

[0041] The exemplary memory 30 of Fig. 1 further includes a web browser 36 for viewing Internet content and a separate collection of items, referred to herein as a client parser application 38. In some embodiments, client parser application 38 uses the one or more references in registry 34 to obtain a login construct from server 50. In various embodiments, in accordance with the present invention, client parser application 38 runs in conjunction with one or more software modules, such as an event log module 98, which tracks user activity, a message interface module 107, which serves as a communication interface between the client parser application 38 and web server 58 and/or external web servers, an application media processing module 109, one or more Application Media Packages 104-1 to 104-n, and an Application Media Viewer 119 which, among other functions, regulates the characteristics of visual manifestations of Application Media Packages 104-1 to 104-n when displayed on output device 42.

[0042] Server computer 50 includes standard server components, including a network connection device 46, a CPU 52, a main non-volatile storage unit 54, and a random access memory 56. Further, server computer 50 includes one or more internal buses 48 for interconnecting the aforementioned elements. Memory 56 stores a set of computer programs, modules and data to implement the processing associated with the present invention.

[0043] The embodiment of memory 56 illustrated in Fig. 1 includes a web server 58 for processing requests received from client computer 20. Web server 58 has many components, including a variety of modules and data structures to assist users that want to log into system 10. Namely, login module 60 handles an entry request from a client

computer 20 and accepts a login identifier that corresponds to a user from client computer 20.

[0044] Once a user has successfully logged into system 10, request server module 72 handles requests for specified Application Media Packages 104-1 to 104-n from client 20. When such a request is received, request server module 72 routes the request to an address that corresponds to the specified Application Media Packages 104-1 to 104-n and transmits the specified Application Media Packages 104-1 to 104-n to client 20. One class of specified networked information handled by request server module 72 is requests for Application Media Packages 104-1 to 104-n. When such a request is received, request server module 72 searches Application Media Packages database 74 for the specified Application Media Package. Application Media Package templates database 74 includes a large number of Application Media Package templates. Each Application Media Package template defines the characteristics of a specific Application Media Package, including fully configurable frame characteristics, viewer and control characteristics, and Application Media Package content references.

[0045] The web server 58 illustrated in Fig. 1 further includes additional modules 131 to handle specialized features of the present invention. For example, one embodiment of the present invention provides a mechanism that allows users to distribute Application Media Packages to each other. In such embodiments, a special server module 131 provides instructions for storing the Application Media Packages, which are to be distributed, using a sharelink database 78. Advantageously,

Application Media Packages that are distributed to other users are customizable. A user can, for example, resize and position a particular Application Media Packages prior to sharing it with another user. Indeed, it is possible, in such embodiments, for a user to arrange a series of Application Media Packages in a unique arrangement and then distribute the collection of Application Media Packages in the designated arrangement. As an illustration, a user arranges a first Application Media Package that represents a scrolling stock ticker at the bottom of an output means, such as a computer screen, a second Application Media Package that tracks the NASDAQ top ten most heavily traded stocks in the upper left corner of the output means, and a third Application Media Package that tracks headline news on the upper right hand corner of the output means. Then, the user distributes the three Application Media Packages in this customized arrangement to other users. Observe that in this example a user of a client computer is aggregating separate sets of information in different Application Media Packages. This stands in contrast to prior art approaches where a web server running on a server computer aggregates information in a single page.

[0046] System 10 is highly scalable and thus supports a large number of users. This scalability stems from the fact that the server 50 is delivering the definition associated with an Application Media Package. The content displayed in the Application Media Package may be located on a separate computer. Memory 56 may provide a statistical analysis module 133 for tracking key events associated with users. This information is stored in statistics database 80. The information collected by statistical analysis module 133 is used for a wide variety of purposes, including server load

optimization and directed advertising, as discussed below. As described below, the statistical information gathered in accordance with the invention includes fully traced events defining the type of content and the duration over which all content is viewed by a user. This type of comprehensive information is not available using present techniques. Much of the distributable computer readable media that is available for processing is stored as content elements 94 on server 82. Server 82 is a standard web server that includes components such as a network connection device 88, a CPU 86, a main nonvolatile storage unit 84, a random access memory (RAM) 92, and one or more internal buses 90 for interconnecting the aforementioned elements. RAM 92 includes some of the content elements 94 stored by server 82. Other content elements 94 are stored in storage unit 84. In some embodiments, a single web server 58 is capable of directly accessing content elements 94 located on one or more servers 82. In other embodiments, each server 82 has a resident web server module that works in conjunction with server 50 to identify, optionally dynamically generate, and serve content elements 94 upon demand.

[0047] With the general architecture of a system within which the present invention may operate provided with reference to Fig. 1, we next turn to a number of definitions of key terms used herein. Terms not specifically defined herein shall be understood to have their broadest, generally accepted meaning. Other terms may be defined elsewhere in the present disclosure. (See also Appendix A and B herein.)

[0048] Application Media Package – An Application Media Package, also referred to herein as a Dot, is computer software component, such as XML code and

data, representing the definition of a client-side mini-application, which displays information and/or provides functionality to an end user. The Application Media Package combines the packaging, application behavior, and the presentation of Internet content with the content itself, creating one integrated unit - a Dot. Therefore, Dots are the integration of application and media. A Dot may be viewed as a free-floating mini-site. It is frame in which Internet content is presented (although lacking the most common functionality typically associated with a browser, navigation).

[0049] In one embodiment, Dots are programmed with browser readable language that is parsed to the Microsoft Internet Explorer (IE) rendering object (referenced within a control, so named the Dot Web Conduit) and are capable of displaying any Internet content supported by IE 4 (however, not necessarily requiring use of Internet Explorer or any browser for such display). This browser readable language defines the appearance of a Dot, its functionality, and what content is presents. This browser readable language includes XML, streaming media, dHTML, etc.

[0050] Structurally, the Dot or media application package comprises initialization data, control calls, and a frame. Content, such as that obtained from a network reference, is rendered within the frame. The control calls may include the web conduit, base controls such as "close" and resize", base GUI elements such as a title bar, Dot menu, and Dot bottom bar, as well as XML Dot calls that are made by the Dot Definition and executed by the Home Dot. Unlike a downloadable, custom,

client-application (e.g., for delivering custom web applications to users' desktops), an Application Media Package is entirely content. That is, the entirety of a Dot package, referred to as its definition, is in a browser-readable language.

[0051] All that is required to instantiate an Application Media Package (on the client computer) is its definition assuming that the client computer has an Application Media Viewer (discussed below) installed. That is, no browser is needed to render an Application Media Package. No browser need be running or be present on the client machine in order to render an Application Media Package.

[0052] Application Media Package Content – Application Media Package content, also referred to as Dot Content, is Internet content served off of partner/Dot Developer servers or another third party server. While in some cases a party developing and making a Dot available (from a partner/Dot Developer server) may also host (on the server) content for that Dot, the developer is free to cause the Dot to point to content from another party/site for rendering as part of an instantiated Dot. It is an attribute of the present invention that any internet content can be accessed, processed, and/or displayed as Dot Content or Application Media Package Content.

[0053] Application Media Viewer – The Application Media Viewer, also referred to herein as the Home Dot, is a network enabled, client application that loads and displays an Application Media Package on a client computer. Data contained within the Application Media Package is parsed by the Application Media Viewer and rendered within the extent of the Application Media Package's frame as defined

~~therein. Data is web browser readable language including media and Internet references thereto, as well as control calls. These control calls contained within an Application Media Package are also parsed and executed. The Application Media Viewer is required to instantiate an Application Media Package (i.e., to create a Dot and provide any associated functionality on a client display device). The Application Media Viewer effectively takes the place of a browser application when rendering Internet content via an Application Media Package. However, an Application Media Package contains Internet data and as such, this data may also be referenced by a browser. In the preferred embodiment, a browser is not employed to instantiate an Application Media Package, nor to invoke any functionality they may provide. Also in the preferred embodiment, the Application Media Viewer is a compiled application, programmed using MFC (Microsoft Foundation Class) or the like. The Application Media Viewer need only be downloaded to the client computer one time and may be used to instantiate any number of Application Media Packages. More than one Application Media Package may be instantiated at a time, i.e. the Application Media Viewer is capable of executing and displaying multiple instantiated Application Media Packages. The Application Media Viewer is also capable of checking for updated versions of itself (and any Application Media Package) on the server. The Application Media Viewer supports a number of facilities including support for standard internet protocols (http, https, etc.) Additionally, the Application Media Viewer also supports collecting and organizing Application Media Packages, user login, user activity statistics collecting and reporting, and messaging between Application Media Packages.~~

[0054] Block – A group of Dots. Blocks are created and destroyed by the user through the dragging and positioning of individual Dots or other blocks.

[0055] Computer – as used herein, the term computer is meant to refer to any device that includes a Central Processing Unit (CPU) connected to a random access memory, a network connection, connected and rendering to one or more output devices, which has sufficient operability to utilize Internet data, such as web-enable cell phones, personal digital assistants (PDAs), etc. Thus, the term is intended in its broadest sense to include not only traditional workstations, laptops and similar devices, but web-enabled cell phones, PDAs, etc.

[0056] Content Server – Also called a Partner Web Server, used to deliver Internet content to a Dot on a client machine.

[0057] Controls – A Dot comprises of at least one control, each of which is arranged within the frame according to the Dot Definition. In the preferred embodiment, this minimum control is the Web Conduit which is the Internet data rendering portion of the frame. Controls are calls that are contained within the Dot content that enable a Dot, for example with application behavior amongst other attributes and explained further herein. The Dot Definition may contain base control (close, resize) overrides such that the controls are not rendered or are rendered only upon mouse over events rather than persistent display. Additional media elements and network referenced media may also be contained within the Dot Definition.

~~**[0058]** DoDots - The working product name of the invention described herein.~~

~~**[0059]** Dot Class - An instance of a particular set of binaries to create a particular type of Dot.~~

~~**[0060]** Dot Definition - This software entity contains enough information to instantiate an Application Media Package (a Dot) and pointers to location of Application Media Package content (Dot content). A Dot is defined by its dimensions and a set of four attributes called Dot components, as will be later described. A Dot Definition also defines access to Dot Web Conduit, which is a control element of a Dot and is a fully functional rendering element that can access available Internet content.~~

~~**[0061]** Dot Link - Hyperlink or URL on a server for downloading a Dot. Based on standard Internet links, these point to or reference particular Dot Definitions and enable the Home Dot to quickly access (using standard HTTP/HTTPS requests, for example) the frame and initialization pointers/URLs in the Dot Definition to instantiate the Dot and render it.~~

~~**[0062]** Dot Messaging Architecture (DMA) - Elements within Dots and the Home Dot application create a messaging architecture (the Dot Messaging Architecture) that enables elements of the system including Dots, controls within Dots, and the Home Dot application, to communicate with one another. The DMA enables these elements to exchange information, request actions or functionality, and respond to system, element, or content events.~~

~~**[0063]** Dot Pack - A group of Dot templates specified by a particular Dot developer, provider or aggregator that complement each other in some way.~~

~~**[0064]** Dot Server - Also called an Internet Application Server, used to deliver Dots to a client machine.~~

~~**[0065]** Dot Template - The XML definition of a particular Dot provider's Dot in its default state. Information includes the Dot size, Dot class, starting URL, etc. The Dot template may be defined by a third party Dot provider. The Dot template is used as the starting point for a particular Dot, which may be customized by the user in some way after the first instantiation.~~

~~**[0066]** Frame - The frame, also referred to as a Graphical User Interface (GUI), defines that area upon a display of the Dot instantiation in which data may be rendered. As previously described, the frame may contain rendered Internet data. A Dot is comprised of a frame or GUI, initialization input data including data and network references to data displayed in said frame. The appearance of the frame (if any) together with data that is displayed therein is the visible instantiation of the Dot. Any Internet data or media data may be contained within the frame such as an image, Internet content, etc. Additionally the frame shape and content may be wholly defined by the Dot developer.~~

~~**[0067]** Lead Dot - The controlling Dot within a block. A lead Dot is the only Dot within a group that presents a control and close box. The lead Dot is the leftmost, topmost Dot (in that order).~~

~~**[0068]** Share - A Share is a XML file that represents a collection of one or more Dots and/or one or more Snapshots.~~

~~**[0069]** Snapshot - A point-in-time data set describing an individual user's overall visible Dot configuration. This includes location and configuration information on the user's overall configuration, and all blocks and Dots.~~

~~**[0070]** Web Browser Readable Language - standard Internet content that is capable of being parsed by a browser, such content including HTML, Java script, XML, CSS, streaming media, Flash, HTTPS, cookies, etc.~~

~~**[0071]** Web Conduit - a control that accesses and renders standard Internet content. Dots in conjunction with the Web Conduit control can render or process anything a browser application can render or process. That is, the web conduit is an entity which processes or handles standard Internet content for rendering the GUI (graphical user interface).~~

~~**[0072]** The present invention discloses a technology that is capable of distributing standard Internet content without the use of and restrictions imposed by a browser as detailed hereinabove, in a manner that enables developer control of the user interface. The present invention includes a method and system in which Internet content~~

~~is developed, packaged and delivered from a server. The present invention further describes methods and systems in which the user obtains content, interacts with it, and is presented with updates to the content either in part or including the entire Application Media Package.~~

~~**[0073]** The present invention provides an alternative model of viewing content obtained from the Internet. Unlike the web page model requiring a web browser, the present invention is founded on the premise that the basic unit of the Web and its media should not be limited or restricted to a webpage-based display for presentation in a conventional browser nor should the user's experience of the Web be limited to one page at a time as defined by a conventional browser (such as Microsoft's Internet Explorer™)~~

~~**[0074]** Furthermore, the present invention enables efficient placement of web applications directly onto users' desktops. The present invention enables content providers to: break their Internet-based content up into smaller pieces; package this content as Application Media Packages (Dots); give Dots an application behavior through a Dot Messaging Architecture; and to distribute Dots from one user to another (Viral Distribution Architecture).~~

~~**[0075]** The ability to break web-based internet content into smaller packages (e.g., Dots) for transfer independent of a conventional browse enables content providers to distribute their media to: fully brand and control a user's experience; obtain direct access to consumers; secure longer on-screen presence; give application-media viral~~

characteristics; and eliminate the need to maintain a software application in order to achieve these characteristics.

[0076] With regard to viral characteristics, for example, the present invention allows content providers to enable users to: collect Dots; use Dots in groups (Dot Packs); and share Dots with others. Therefore, Dots are passed from server to user and also from user to user via email links. Not only are may single Dot links be virally distributed, but also collections of Dots and their links may also be virally distributed from user to user. Because Dots reference web data including media, Dots enable the viral distribution of said media although the media may reside on a server.

[0077] Software application download is eliminated because Dot content is based on standard Internet content (HTML, dHTML, flash, streaming media, Java, etc.), providing Dots with certain browser-like functionality. By this virtue, Dots may be distributed not as software application downloads but as Internet media downloads, similar to web page downloads. As a Dot may define the frame within which content is to be rendered, the visual extent of a Dot is not bound within the borders of a browser window. Unlike a web page in a third party browser, a Dot's graphical user interface may be wholly defined and/or branded by the developer.

[0078] Because Dots are instantiated and rendered upon a client computer as separate and independent graphical user interfaces, and because they may be continuously refreshed with new internet content, Dots may persist on a user's screen. Therefore Dots may possess a longer desktop presence (i.e., remain in place, or "On",

~~while a user attends to other tasks on a device, such as interacting with different applications, making phone calls, etc.~~

~~**[0079]** The present invention therefore also allows for persistence of displayed Internet content. For example, a dedicated frame may be rendered on a user's display, Internet content obtained from a server, the obtained Internet content presented in the frame (and possibly periodically updated), and the frame and content positioned and sized such that it may remain, unobtrusively, open on the user's desktop, even when other windows are open. This is fundamentally different than today's model where users visit websites and leave them due to size of the browser window, full use of the content displayed, etc.)~~

~~**[0080]** Dots provide Internet content developers and web application developers: a unique way to package their content as custom client application experiences without having to develop, distribute, and support their own custom client application. Developers may repurpose the Internet content that they have developed and used in web pages to form Dots. In doing so, according to the present invention, developers may also add application behavior to Dots such that the user experience includes interactive graphical user interface elements, likened more to a custom desktop client application rather than to a web page. Thus, Dots enable Internet media to posses application behavior without the use of a browser or custom application.~~

~~**[0081]** From a Dot provider's perspective, Dots provide direct access to users; and a persistent branded presence on users' desktops beyond and without the browser,~~

thus providing full control of the user's experience. That is, a Dot that is instantiated on a user's client computer has the user's display real estate. Conversely, a web page typically must be navigated to through the use of a web browser each time a user views the web page media, and once the user navigates to a new page, the display space (browser window) effectively belongs to the new web page content. As previously described, the branded experience may be wholly defined by the Dot developer, thus removing the rendering and branding limitations of a web page in a browser. Without the present invention, an alternative available to the Internet content developer is to develop a custom application that must be downloaded each time it is changed or alternate content is desired to be displayed.

[0082] In summary, Dots give users a unique way to experience web media by virtue of its: (1) flexible frame definition; and (2) a Dot's persistence when instantiated. Because of their ability to persist on a user's desktop apart from a browser, Dots also provide a convenient method to access content and services (without separate user selection of a URL). With a Dot for each of a user's specific Internet data needs, Dots eliminate the need to download custom software from multiple sites for various content.

Rather than implementing custom client applications in Java, C++, etc., Dots are defined using XML and filled with standard Internet content such as HTML, GIFs, and the like.

[0083] The present invention also provides for the ability to collect and share favorite content and application-like behavior (i.e., sharing Dots). Links to Dots may

~~be easily emailed between friends. Dots may be collected according to common interest, usage requirements or the like and defined as groups called Dot Packs. These Dot Packs may also be shared as links via email or server.~~

~~**[0084]** A fundamental aspect of the present invention is the creation of a Dot, including packaging of Internet content. The goal is to create an Application Media Packages (Dots) which combine the packaging, application behavior, and presentation of Internet content with the content itself (or typically a pointer thereto), as a single integrated unit (a Dot). Therefore, Dots are the integration of application and media.~~

~~**[0085]** This is fundamentally different than rendering internet content in a viewer application or alternatively, developing, distributing, and supporting a custom client-side application. Typically, a Dot definition includes details regarding a window "frame," and the Dot content has access to the entirety of this frame. The details regarding the frame is itself Internet content, so that the entire Dot "package" (including the definition of the package) is Internet content. Thus, the Internet content is not trapped in a third party viewer (e.g, Stock Trading site's web page in Microsoft's browser).~~

~~**[0086]** Dot content has access to application behaviors of the window frame (size, position, look and feel) and of the Dot application system (show other Dots, delete Dots, etc.) For example, a Dot may change color according to ongoing data updates that~~

it receives over the Internet. A Dot designed to display weather information may become increasingly red as temperature data received for a particular city or region increases.

[0087] Furthermore, a Dot may communicate with other Dots. Communication between Dots is typically carried out by two or more Dots that are instantiated on the same client and enabled such that they may participate in inter-Dot communication. In one embodiment, communication is carried out by messages that are passed between Dots via the Home Dot that instantiated them. These messages may pass data, alter controls, or result in behavior change for example. Not only may Dots of the same kind communicate but in one example, Dots formed between different developers may also communicate and affect one another. Such communication may be apparent or may be a background function supporting some aspect of a Dot's functionality.

DOT AND DOT DEFINITION

[0088] With the above general description in mind, we now turn to a description of a Dot and its definition. With reference to Fig. 2, the primary components of a Dot 100 are: Internet content 101, a visual object(s) 102 within which the Internet Content 101 may be rendered, and operation element(s) 103, which perform certain functions, each discussed further below.

[0089] A Dot 100 is an instantiation of a Dot Definition 104, the components of which are illustrated in Fig. 3 At its broadest, Dot Definition 104 comprises an XML (or similar) definition 105 (which may include initialization data, control calls, and a

~~frame definition, discussed further below) and a Network Reference 108 (e.g., a URL) to Internet content 101. (See, for example, Appendix A - DoDots XML Specification.)~~

[0090] ~~XML definition 105 includes an appearance definition 106 for defining the appearance of a GUI (discussed further below), and a control definition 107 for defining controls associated with the GUI. Definitions 106 and 107 will typically include initialization data, control calls, and other elements. Optionally, a Dot may include message data 110 (e.g., access to the Dot Messaging Architecture). Application Media Package 104 (or "Dot Definition") may also include tags 112 to identify the Dot.~~

[0091] ~~The XML definition 106 of the appearance of the GUI typically includes a frame definition, specifying size, location, etc. Fig. 4 is an illustration of an instantiation of a generic GUI 160, which comprises a frame 162 typically divided into a number of control regions. For example, Frame 162 may include a title bar 164, a bottom bar 166, a menu control region 168, and other definable control regions 170, 172, and 174. One critical control region, referred to herein as the Web Conduit 176, is a regions in which many Dots will present Internet content. Returning to Fig. 3, definition 106 is responsible for providing the dimensions and general shape of the GUI.~~

[0092] ~~The control definition 107 defines the layout (arrangement of the controls) and functional routines or pointers to functional routines (what the controls do when accessed by a user). Control definition 107 may also include XML Dot calls that are made by the Dot Definition and executed by the Home Dot (not shown here).~~

[0093] Network reference 108 will typically be a URL containing the address of a server having stored thereon data (Internet content) for retrieval and display within the GUI. Functionality typically provided by an associated Application Media Viewer controls the accessing of the server at the URL and the retrieval and processing of the Internet content for display. Importantly, retrieval and display of the Internet content may take place without resort to a Web browser application. Indeed, no Web browser application need be executing or even present on the client to support the Dot functionality (although embodiments in which Dot functionality is operational in conjunction with browser functionality is within the scope of the present invention). Internet content obtained from network reference 108 may be rendered within the frame defined by and according to the layout definition dictated by the frame appearance definition 106. Internet content obtained from reference 108 may itself include XML calls providing certain functionality.

[0094] As previously mentioned, Dot Definition 104 is typically XML code. These definitions are quite simple to author, and to edit if needed. The definition is content, rather than compiled code, which provides additional flexibility at the client side should it be desired to modify the definition (e.g., allow a user to edit the network reference). As will be described further below, when an Application Media Viewer is operating on the client computer, the Dot is effectively both content and instructions. This is therefore a method of packaging internet content as an application (as compared to a hard coded custom client application) for operation on Internet content.

~~**[0095]** Fig. 5 illustrates a rendering or instantiation of a Dot Definition as a Dot 120, in this case a visual indicator of likelihood of rain, expressed as a percentage 122, and rendered against an image of a cloud 124. According to the above, a definition for Dot 120 includes a definition of the size, configuration, and location of a first control space 126, a specification of the layout and functionality of control interfaces (or simply controls) located in a second control space 130. Examples of such controls include a "close" button, a "resize" handle, etc.~~

~~**[0096]** The image of a cloud 124 against which the likelihood of rain is rendered is a static image. The data representing the instructions for rendering the cloud may be found in the appearance definition 106, and thus resides within the Dot Definition 104. Alternatively, the data representing the image of the cloud 124 may be obtained from the location to which network reference 108 points. In either case, the data representing the cloud is utilized by the Application Media Viewer 104 to render the cloud image.~~

~~**[0097]** The actual data representing the likelihood of rain may be standard Internet content, and will change periodically as controlled by the source (e.g., the Internet content itself obtained from a server at network reference 108) of the data. Since this data changes as controlled by the source, the data is considered dynamic. This dynamic data is obtained by the Application Media Viewer from the source, and rendered atop the static image of the cloud. The frequency of updating and re-rendering of the dynamic data may be under the control of the Dot Definition or the source of the data, as a developer deems appropriate. The data to be rendered with the image may for example overlay, or be rendered in a layer on top of the image, beside the image, in the~~

same layer or otherwise, in a layer under the image layer, etc. However, there is generally an intended relationship between the position of rendered Internet content and other items rendered in the graphical user interface. For this reason, we say that the Internet content is rendered "in association with" the image within the graphical user interface.

[0098] It will be appreciated that Dot 120 is merely on illustration of a Dot, and many other types of content, format, layout, controls, functionality, etc. are contemplated by the present invention. For example, additional display of information and/or control may also be presented in top/bottom bars 132, 134, respectively, as will be understood by one skilled in the art.

[0099] Thus, a Dot is defined using, for example, an XML file which is the embodiment of a Dot Definition 104. The definition contains enough information that, when instantiated, the Dot may be rendered and filled with Internet content from a source. The Dot Definition contains data used to define and configure a frame and its elements, specify and lay out the controls, and specify parameters that initialize all the Dot's components with content and data.

Frame Definition

[00100] A frame can be defined to have a configuration (e.g., base shape), size (e.g., dimensions), and initial location (which may be changed by a user upon instantiation). In additional, a simple Dot may be defined to have four default

functional frame definition controls, for example to encourage user interface consistency between different Dots. These four default functional controls include a title bar 132 (Gif rendered with title bar properties), a Dot menu 136 (with flexible menu entries), an exit control 128, and a bottom bar 124 (Gif rendered with bottom bar properties) with corner elements 138, 140 (for sizing and consistent user interface). A default layout of these functional components may be set, such as positions for the title bar 132 at the top of a Dot (Dot Menu 130 on its left, exit control 128 on its right) and the bottom bar 134 at the bottom of a Dot (with corner elements on either side). It will be appreciated that this is one form of layout, of which many others are within contemplated within the scope of the present invention.

Title bar

[00101] The title bar 132 may contain a reference title for the Dot and may provide for a place for a user to grab and drag the Dot in a windowed environment. It may be implemented as a GIF rendering control that can be targeted to a local/remote title bar 132 image (an embodiment supports four title bar images - normal, mouse-down, mouse-over or hover, inactive). In this embodiment, the title bar 132 has a fixed height and width that is a function of the frame's width. The Dot Definition allows the title bar 132 image to be justified left, right, or center and for specified number of repeatpixels, the title bar image may be tiled the full width of the Dot. Overlay text can also be specified to layer on top of the title bar 132 image.

[00102] The default size and the default position of the Dot are defined as part of the appearance definition 106 in the Dot Definition 104. The Dot size can be specified in pixels, or Dot Units (1 Dot Unit is 40 pixels), etc. Dot Units can define a grid that keeps Dots sized on the same units so that they more easily and neatly align as well as cleanly snap to each other when Dots are used together. Dots can be sizeable or fixed-size in either dimension or both. The default screen position for a Dot can also be specified in pixels, or in relative position - top, left, center as provided for in the appearance definition 106 in the Dot Definition 104.

Menus

[00103] One embodiment of the present invention reserves the four corners of a Dot for functional branding elements. The upper left comer is currently reserved for the menu control 136. The upper right corner is currently reserved for the Dot exit 128 control. The lower corners 138, 140 are for sizing the Dot if the Dot is resizeable. Menu entries for branded menu control 136 (upper-left) can be specified in the Dot Definition 104, for example at 112. The menu items can be named, assigned images and tooltips, and assigned a specific and targeted action, for example a DotMessage 110 sent to a specified element (address). (See DMA messages hereinbelow). The Dot exit 128 button (upper-right) closes the Dot. An on-close action (such as a message and address pair) can be assigned to the Dot close event.

Bottom Bar

[00104] The bottom bar 134 may be implemented much like the title bar 132, including supporting text overlay and additional functionality. (See Appendix A – DoDots XML Specification for a list of controls with XML call and their definitions)

Alternative Embodiments for Controls

[00105] Dots may be defined so as not to require a rectangular title bar 132 or bottom bar 134 (at top and bottom). Indeed, either or both of title bar 132 and bottom bar 134 may be omitted in a Dot Definition. Other controls (128, 130, 138, 140) may also be omitted, provided certain functionality (e.g., exit or close) is otherwise provided. Furthermore, every functional element in a Dot may be defined as a control element that is relatively positioned and layered (in z order) – much like layers in HTML. Transparency may also be a given property. Different layers and controls may be given Dot properties (such as a title bar move property or Dot menu property). Very flexible Dot interfaces may therefore be provided. (See the specification for the <DOT> and <CONTROL> DTDs in Appendix A – XML Specification.)

Control Space

[00106] Between the title bar 132 and the bottom bar 134 is a control space 126 where one or more controls, images, data, etc., can be flexibly positioned. For example, control space 126 may include a web rendering control referred to herein as a Web

~~Conduit (described further below). The Dot framework supports any Active-X based control which may be positioned and initialized in a Dot (e.g., an on-line stock trading company may implement a custom stock chart-rendering control and define a Dot to permit a user to interface with this control).~~

~~[00107]~~ ~~A Dot may be defined to include a static image over which dynamic data may be displayed, such as the example of Fig. 3 in which a static image of a cloud has rendered thereover dynamic (changing) data relating to likelihood of rain. Alternatively, the Web Conduit may render in control space 126 purely static data obtained from the Dot Definition, a URL, or the client device, as well as dynamic data resulting from client-run processes, pushed or pulled Internet content, etc.~~

~~[00108]~~ ~~The Web Conduit control is just one of many controls that may be included and positioned in a Dot's control space 126. For example, Active-X based control can also be referenced and inserted. This capability allows a Dot developer to implement a custom control. For example, a stock ticker display may be implemented as a custom secure chart control that renders stock tracking charts with small streams of secure xml data. Such a custom control can be laid out with a Web Conduit control next to it if the Dot developer chooses to do so. Also, just like a browser, a Dot developer may embed an active-x control in an HTML page rendered in the Web Conduit control as is the practice for use with full screen browsers.~~

Web Conduit Control

[00109] The Web Conduit (mentioned above) control can render Internet content in a fashion similar to Microsoft's Internet Explorer™ (IE). This control functionality may be provided, for example, by utilizing Microsoft's Internet Explorer's (1E4+) WebBrowser object. Note that the 1E4+ object does not constitute a browser. The Web Conduit merely uses rendering tools of the object - it does not invoke or require the Internet Explorer application or any other web browser application. Rather, functionality is provided to support rendering of Internet content so as to integrate this control with the Dot framework and to receive and transmit messages in and out of the HTML rendered in the control.

DOT IDENTIFICATION

[00110] A Dot can be identified (in the Dot Definition 104) by three ID strings: GlobaliD, Domain, and Kind reference. These are defined as follows:

GloballD - A Dot's GloballD is used when a Dot Definition is within a Share (described further below); this ID is unique with respect to other Dot tags in the Share.

Domain - A Dot's Domain is a unique label for the owning company Dot developer of the Dot.

Kind - A Dot's Kind (specified by the Dot developer) is a helpful identifier for finding the Dot; A Dot's Kind does not have to be unique.

CATEGORIES

[00111] In an embodiment of the present invention, categories are used as an organizational tool. A Dot Definition 104 (Fig. 3) may be provided with an element 150 defining the category(ies) with which that Dot is associated. A mechanism is provided to allow identification of categories of Dots, useful for selecting, sorting, organizing, etc. The categories that a Dot belongs may be edited by editing the string elements in the Categories element 150 of the Dot Definition 104. (See the specifications for the <ALLCONFIG> and <DOT> DTDs in Appendix B.)

EVENTS

[00112] A Dot Definition may also include an Events element 152 in which actions can be assigned to certain Dot Events such as the Dot's ONCLOSE event. DotEvents can be assigned a specific and targeted action much like a menu control 136 item; currently this action is implemented as a DotMessage sent to a specified recipient (see DMA messages, below). When the specified DotEvent occurs, for example ONCLOSE when the Dot closes, the specified message is sent to the specified recipient.

~~**HOSTING AND SERVING DOTS**~~

~~**[00113]** Referring again to Fig. 1, web server 50 may serve the Dot Definitions 104 as an XML file to client computer 20. The Dot Definition 104 may be served elsewhere such as a third party or partner server (not shown) along with the Dot content that fills in the Dot. In one embodiment, this XML specification may be kept proprietary and Dot developers define and package Dots indirectly without having to author XML Dot Definitions. In another embodiment, the xml specification is open, and content providers (and others) have complete control of the authoring of Dots.~~

~~**[00114]** Dot Definitions 104 are indexed and stored in server 50 in database 74, and are accessible to Dot-rendering and Dot-management applications (the Application Media Viewer) via "DotLinks". These DotLinks, based on standard Internet links, point to particular Dot Definitions and enable the Application Media Viewer (using standard HTTP / HTTPS requests) to quickly access the frame and initialization pointers or URLs in the Dot Definition to instantiate the Dot and render it, filling with DotContent (Internet content served by a partner Dot Developer servers 82).~~

~~**[00115]** Dot developers (e.g., content and/or service providers) serve the Dot content obtained from source or reference 108 (the standard internet content that fills in the Dot). According to one embodiment, Dots may be served my numerous different non-proprietary servers 50. Content may also be provided by a non-proprietary server, as specified by the Dot developer. Of course, it is possible that in other embodiments specific Dots may only be obtained from certain sources.~~

[00116] Because the Dot Definition 104 contains enough information to instantiate the Dot as well as the reference 108 that address the location at which the Dot content is located, a Dot is easily and quickly distributed, as well as collected and shared by users. (This is discussed further herein with regard to Viral Distribution Architecture.) It is therefore easier and faster to get information delivered in a Dot than in a web page.

[00117] This Dot-based architecture is very different than the current model of Internet content. Today, users visit web sites, following a destination-based model of content access. The process of accessing media therefore is sequential or linear, resulting in user viewing of one full-screen web page at a time. Furthermore, there is no simple to use or inherent mechanism in place today to share sites short of sending URL's.

ENHANCED DOT CONTENT

[00118] The Dot Definition 104 code is accessible, flexible and dynamic, enabling it to be modified at any time, even after it is rendered. For example, Internet content 101 (HTML) can access and modify its associated Dot Definition 104 by sending messages to other portions of the definition, such as the definitions of appearance 106 or control 107. Other application system-level functionality is also available by sending messages to the Application Media Viewer 199, discussed further below.

~~**[00119]** By using an architecture, referred to as Dot Messaging Architecture, for communication between Dots, and between a Dot and the Home Dot, standard Internet content can access and enable Dots to exhibit application behavior. That is, content can provide functionality typically associated with applications, such as dynamic refresh, contextual presentation, interactive response between user actions and changes in the graphical user interface or frame, etc. (See "Dot Messaging Architecture" below.)~~

~~**EXAMPLES**~~

~~**[00120]** The following examples are used to demonstrate key features of the present invention Some of these features include: a new way to view standard internet content, a fully branded experience for the user, and a web experience that delivers true application behavior. Sample companies are used in these examples to demonstrate branding together with the use of internet content to produce a novel user experience that transcends the web page. Trademarks are those of the respective companies.~~

~~**E*TRADE**~~

~~**[00121]** E*Trade's primary business is to enable users to execute securities trading online. Therefore stock information and notification are essential elements to their business, as are the transactions themselves. Persistent display is an important aspect of Dots as is the ability to provide ongoing data updates. Below is a list of aspects that illustrate novel advantages that Dots provide for a possible E*Trade Dots system.~~

- E*Trade can leverage their existing content through Dots (Dots are built on top of standard HTML]

- E*Trade can break their content into smaller pieces and package them into Dots.

- E*Trade can give their Dots application behavior. Instead of being trapped inside a viewer window, content providers have access to the entire frame, which enables them to brand the Dot, access to application and system features. Content providers can also enable users to resize the Dot click, to open E*Trade Dot, ability to execute operations in the Dot such as trading stocks as opposed to only monitoring stock prices.

- Because Dots can be left on or actively running and displayed on a client computer, they can notify users of coming/pending/immediate content events. A Dot can notify a user in many ways, including: popping up a message box or another Dot. A Dot can resize itself and show new content and/or bring itself into focus, as a window application may also do.

MERRIAM-WEBSTER

[00122] In this example, Miriam-Webster has word definition content that is useful for users to access. Users may be better served by the Dot format of presenting as opposed to traditional web pages. Below is a list of features that illustrate this point.

~~Miriam-Webster can leverage their existing content to provide a dictionary "application" in a Dot; the content is HTML delivered from their servers. In a Dot, Miriam-Webster can provide direct access to just the information users are interested in viewing without requiring them to disengage in other operations such as word processing work in order to actively seek information from MiriamWebster's servers. Additionally, content providers can add new functionality to their Dot applications just by changing the HTML files on their servers that are delivered to Dots or as Dot Definitions 104. They do not need to revise and redistribute custom client software.~~

~~EBAY~~

~~[00123]~~ ~~A single company such as eBay can use multiple Dots to engage users from many points at the same time to facilitate a more complex process, such as online auctions or shopping. Many companies have an enormous amount of assets (content, services, and applications) that can only be exposed to users one page at a time. Featuring and exposing new or important functionality is often difficult to do if they are buried several pages deep.~~

~~Dots enable content providers to feature and expose functionality and services in a new form, leveraging and featuring all of a company's assets in a consistent way. eBay, for example, might provide a "MyBids" Dot that allows users to more closely follow those items they have bid on (the current asking price and when the auction will expire).~~

~~eBay can also provide a Gallery Dot. Instead of wading through thousands of thumbnails in the gallery section to find what their looking for, users can identify categories they are interested in and window shop these items, click on a particular item to add new item to MyBids Dot. When a user finds something that interests him, it's added to the MyBids Dot with a single click.~~

~~There is a messaging architecture (DMA) that may be enabled by the Dots architecture disclosed herein which enables Dots to communicate with each other, thus facilitating operations between Dots without user intervention. For example, a user finds an item of interest in a "Gallery" Dot. He clicks on that item and it is added to the "MyBids" Dot with a single click. Additionally, a "BidDot" may be caused to pop up for that single item, allowing a user to directly bid and monitor the auction process. He clicks on a second item in the "MyBids" Dot and another "BidDot" pops up.~~

~~MP3 DOT~~

~~[00124]~~ ~~By using an embedded MP3 playing control, an MP3 playing Dot can access, play, and manipulate MP3 files and playlists from a local hard drive or the web, just like a custom MP3 client application. The significant difference is that the GUI & controls (the Dot Definition 104) are entirely Internet content (HTML, javascript, etc.) and can change simply by modifying HTML files online.~~

MOBILE/OTHER PLATFORMS

[00125] Dots can be viewed as a smaller format package for internet application media. This package is more portable than executable applications as well as full screen formatted media for browsers. Portability across platforms requires consideration of diverse screen sizes and resolutions as well as operating systems and user interface controls. This diversity requires companies or users who require cross platform media distribution to scale down their existing content for portability to mobile devices. No such scaling down is required when deploying Dots and the Dot architecture.

APPLICATION MEDIA VIEWER (HOME DOT)

[00126] As previously mentioned with regard to Fig. 1, a client computer 20 will ideally have stored thereon a software component referred to as an Application Media Viewer (or Home Dot) 119. As Application Media Viewer 119 performs a number of functions in the process of acquiring, instantiating, modifying, etc. Dots, in addition to simply viewing Dot content, we also refer to an Application Media Viewer as Home Dot herein. The Home Dot 119 is effectively a client application that contains the procedures or calls to procedures for rendering and managing Dots on the client computer. It therefore has attributes of an application. The Home Dot is designed to operate in conjunction with a Dot Definition, and vice versa.

[00127] Data contained within an Application Media Package (Dot) is parsed by the Home Dot. Control calls contained within the Dot are also parsed and executed.

~~The Home Dot may then instantiate a Dot without relying on functionality provided by a browser. In fact, in a preferred embodiment, a browser is not employed to instantiate or operate an Application Media Package.~~

~~[00128]~~ ~~The Home Dot is a compiled application, and can be programmed using, for example, MFC (Microsoft Foundation Class) or the like. The Home Dot need only be downloaded to the client computer one time and may be used to instantiate any number of Dots. The Home Dot is capable of executing and displaying multiple Dots simultaneously.~~

~~[00129]~~ ~~The Home Dot may be provided with the ability to automatically or manually check for updated versions of itself on a server. If a newer version is detected, the user may be provided with the opportunity to download and install the updated version of the Home Dot.~~

~~[00130]~~ ~~The Home Dot supports a number of functionalities, including support for standard internet protocols (http, https, etc.) Additionally, the Home Dot supports collecting and organizing Dots, user login to a Dot server, user activity statistics collecting and reporting, and messaging between Dots, as discussed further below.~~

~~NETWORK DISTRIBUTION OF DOT AND HOME DOT~~

~~[00131]~~ ~~Referring now to Fig. 6, a communication flow diagram 299 is shown for an exemplary system enabling the distribution and use of Dots between a client~~

computer 199, a server (Dot server) 153 making available one or more Dots including a Home Dot, and content server 155, each with a network address or IP address. It will be assumed for this particular example that a Home Dot 200 has previously been downloaded and installed on client computer 199. It will also be assumed for this example that Dots 120 and 230 have been previously requested (the mechanism for this request is described further below).

[00132] Accordingly, Home Dot client application 200 utilizes Dot Definitions to instantiate Dots 120, 230. There may be ongoing communication 250, 260 between Home Dot 200 and Dots 120, 230, respectively. This communication may include data for the presentation of Internet content, messages, and/or state information. Furthermore, the Home Dot client application 200 executing on a client computer 199 communicates with Dot server 153 through a channel 210 (such as a broadband Internet connection) as well as Content server 155 through channel 157. Internet content is received by Home Dot 200 and rendered into the instantiated Dots 120, 230, as appropriate. The Internet content may be one or more of many formats, such as XML, HTML, GIF, Streaming Media, Flash, HTTP, HHTP(S), etc.

[00133] The Dot server 153 is communicatively connected to a physical memory device 201 which holds a database 202 containing software objects for downloading to or access by a client device such as client computer 199. This physical memory device 201 may be a RAID hard drive system, a standard hard drive, removable media, or any other type of volatile or non-volatile memory known in the art. Database 202 may contain one or more Dot Definitions 104, the Home Dot client application 200,

available for download to the client device, as well as user account data 205, state data including use statistics 206 and Dot index/shares 204 (each described in further detail herein).

[00134] The content server 155 is communicatively connected to a physical memory device 261 which holds the Internet content 262 as well as other forms of data 263. Content/data 262, 263 is communicated to the instantiated Dots 120, 230 where it is rendered on the client computer 199. In general, content passes first through the Home Dot application 200 or a similar client computer 199 program that is capable of receiving and parsing Internet content. Therefore, the connections between content server 155 and Dots 120, 230 are shown as dashed lines, indicating that for this figure the connection is indirect.

[00135] Unlike a downloadable custom client application, the Dot Definition 104 is comprised entirely of Internet content in a web browser readable language. To instantiate a Dot on a client device or computer, only its definition is required (assuming that the computer is Dot-enabled, i.e., that the Home Dot client application 119 has been installed). The Dot Definition 104 contains just enough information to define, layout, and initialize a Dot's components (frame, controls, etc.); this information configures the graphical user interface which may then present Internet content therein. Consequently, a Dot Definition typically has a small file size (~2KB), and is therefore a quickly accessible and loaded XML file.

[00136] One aspect of software distribution according to the present invention may proceed as illustrated in Fig. 7. A user first requests a Dot, or more specifically its Dot Definition 104, by clicking on a Dot link 321 which may be a hyperlink on the web page of a Dot server 153, a web page of a partner or content server 155, or selectable menu item on the client computer 199. A Dot link 321 is an Internet link to a Dot Definition 104. A Dot link operates in a manner similar to an HTML link. One added feature associated with a Dot link is that the server that serves the Dot Definition 104 typically has a sensing mechanism that can determine whether or not a user has the Home Dot application 200 installed on his computer or device 199.

[00137] If the user does not have the Home Dot application 200, then it is downloaded and installed with a first set of Dots (e.g., 120, 230) that may be user-selected or part of an initialization set. This download process is discussed in further detail below. From that point forward, the client computer 199 is Dot-enabled and does not require a subsequent download and install of the Home Dot. In short, only a single application-like software product need be downloaded and installed on a client device to enable a variety of different Dots, as opposed to requiring a user to download a different custom client application for different types of Internet content.

[00138] Once a client machine 199 is Dot enabled, (Home Dot application 200 installed), Dot 120 may be rendered by simply clicking on Dot link 321. This rendering is typically faster than it takes to load and render a small web page. Furthermore, once the Home Dot application is installed on the client no additional executable software need be downloaded or install to view and interact with a Dot. The Home Dot

application 200 accomplishes this by retrieving the Dot Definition 104 specified by the Dot Link 321, instantiating the Dot 120, obtaining the Internet content specified by the Dot Definition 104, and presenting the Internet content (standard Internet content from DoDots' partner servers 155) within the Dot.

[00139] Once a user has received a Dot Definition 104, the Home Dot application 200 remembers the Dot's Dot Definition 104 as part of a user profile 310 so that it can instantiate the Dot and begin filling it with Internet content immediately. Because a Dot Definition 104 becomes part of a user profile 310, it can be modified by use (e.g., the image of its default title bar 125 may be changed, its Web conduit control may be navigated to a different URL using DMA messages, etc.) and the Dot will initialize from its last state stored in user profile 310 the next time the Dot is to be instantiated (or "turned on") at 324.

[00140] A Dot Link 321 represents a Dot. Therefore, this Dot Link 321 (and hence the Dot 120) can be easily and instantly distributed; the Dot Link 321 can be posted on web sites to promote the Dot, or shared with friends via email. This is viral distribution of a Dot, via distribution of the Dot Link 321 – the Dot server hosts and serves the Dot Definitions 104 to which the Dot Links 321 point. Dots may also be aggregated into packs by content providers, or other aggregator, grouped for example by like subject or perceived common user interest, and the packs of links acquired by a user at 322 as if they were single Dots. A user may also provide others with access to Dots when the actual Dot Definition is located on a remote third party Dot server. This may be accomplished by way of a Sharelink, which may be provided by a user at 302.

~~Others may then access the Dot, including setting changes that may have been made by the user, by accessing the Sharelink provided at 320.~~

~~**[00141]** From the client perspective, one example of a process of acquiring and instantiating a Dot is illustrated in Fig. 8. According to this process, a Home Dot is employed to acquire a new Dot, however it is within the scope of the present invention to acquire new Dots through other mechanisms, such as through a web browser application (with utilization of the Home Dot ultimately required in order to instantiate and populate the Dot). After logging in at step 240 the user clicks on the Dot Link signaling a request to acquire the Dot. The Home Dot application then sends a request to download the Dot to the Dot server 153 at step 241. Alternatively, the user may click on a Dot link before logging in, for example if the link has been e-mailed to the user, and then, after clicking on the link, log in. In another aspect, the user could view, but not collect, a transient Dot without ever logging in.~~

~~**[00142]** After the user is logged in and has clicked on the Dot, the Dot server 153 retrieves the Dot template from its Dot database, and provides it to the client in step 243. The client receives the Dot template from the Dot server 153 in step 244.~~

~~**[00143]** For display of the new Dot, the Home Dot creates a frame in the display of the user interface (42, Fig. 1) in step 245. In step 246, the client 199 requests the necessary Internet content stored at the URL(s) identified in the Dot Definition from the corresponding content server 155. The content server 155 transmits the content in step 247, and in step 248 the Home Dot presents the content in the viewer. As the type~~

~~of content may vary greatly, the content may appear within the frame (such as text or images) or may be independent of the frame (such as audio) but the frame including controls for controlling aspects of the presentation of the content. In one embodiment, Internet content is presented such that it is enclosed by the frame, allowing the user to preview the Dot. Alternatively, the Home Dot may simply collect the Dot, adding it to the user's processed user profile 310.~~

[00144] ~~The user may then view the Dot on the user interface display 42, and may interact with the Dot much in the same way as a user may interact with typical Internet content or web applications. This may change the Dot from its present, "raw" state to a used state reflecting alteration or use of the Dot by the user. Thus, the user has stored on client 199 a Dot Definition rather than a Dot template. For example, the user may direct the Dot to display different content within the Dot if the Dot content enables the user to do so. Or, the user may provide information to the content server 155 which allows the Dot to be personalized. The user may additionally be given the option of changing the size or location of the frame, etc.~~

[00145] ~~If the user collects the Dot, the Dot will be added to a user's list of collected Dots. For example, the Home Dot may add the Dot's definition to a processed user profile (discussed further below). Thus, the Dot's "state" will be preserved. Alternatively, the Home Dot may collect the Dot automatically, without waiting for a user command, by adding the Dot Definition directly to the processed user profile.~~

~~**[00146]** If the Dot's state has been altered by the user or by the content - if, for example, the user has directed the Dot to Internet content other than the initially-displayed content, provided personalizing information, or changed the properties of the frame, or if the content itself has caused an alteration in the Dot - this alteration will be reflected in the Dot Definition stored in the user profile. Information which personalizes the resulting content, instead of being stored in a "cookie" on the client device, can be stored as part of the Dot Definition. This advantageously permits personalization of content, such as Internet content that is associated with the Dot content and the user, without storing a cookie on the client 199. One advantage this provides is that the state of a Dot can be returned for a user no matter which computer the user accesses the dot from.~~

~~**[00147]** A user may also access a Dot which has been previously collected, and possibly altered by use as explained above. As previously described, the user profile includes Dot Definitions for Dots which have been viewed and collected by the user. A screen shot showing Dots 502, which have been previously collected by a user is shown in Fig. 9. Also shown are Dot Links 504 available from a Dot server for download to the client.~~

~~**[00148]** One embodiment of the steps taken to provide the user with Dots which have been previously collected are shown in Fig. 10. According to this process, acquired Dot Definitions are maintained on a central server, and a user profile identifies which Dot Definitions are associated with particular users. This is useful when a user~~

may be using more than one computer and wishes to have access to her collection of Dots on any machine she is using. However, it is equally within the scope of the present invention that Dot Definitions may remain resident on a user's computer and not be stored for that user on a central server or the like.

[00149] As discussed above, on login at step 250 the user's profile is retrieved by the Home Dot stored on the client (step 251 and 252). The user's profile, stored in the user profile database, includes the Dot Definition for each of the Dots previously collected, and possibly altered, by each user. The Dot Definition, as discussed above, includes the Dot frame definition and the definition of the controls for filling the viewer within the frame with content. After log in, a local copy of the processed user profile is stored on the client 199, and this copy is further processed as the user collects new Dots, or uses new or collected Dots such that the Dots are altered.

[00150] When the user clicks on the name or icon of a collected Dot at step 253, the Home Dot creates a frame in the display of the user interface (40, Fig. 1) in step 254. At step 255, the Home Dot causes the client 199 to request the Internet content from the URL identified in the Dot Definition from the corresponding content server 155. This content is provided in step 256. It will be appreciated that the URL need not be the same as the initialization URL in the Dot template stored in the Dot template database 202 on Dot server 153, and in fact the content server need not be the same content server corresponding to the initialization URL. In step 257, the Home Dot places the content in the Dot frame, and the Dot is then fully instantiated with content.

HOSTING DOT DEFINITIONS

[00151] Dot web servers 153 host and serve the XML Dot Definitions 104 and provide the Dot Links 321, Dot Packs 322, and sharelinks 320 that point to the Dots so that a user can easily and instantly add the Dot 104 to their Home Dot application 200 (adds Dot 104 to their user profile 310) simply by clicking on the Dot Link 321, pack link 322, or sharelink 320. The Home Dot application 200 registers with the local browser/computer so that Dot Links 321 are handled by the Home Dot application 200 (and not by the browser).

Dot Templates and the Dot Index

[00152] New Dot Definitions 104 get published to a Dot index 204. A Dot Definition can be modified by a user once downloaded. Since these new Dot Definitions are resident on the server for downloading, they are also referred to herein as Dot templates. The Dot index stores Dots definitions or templates 104 in database 202 on Dot server 153 by category (such as sports, finance, games, etc.) with descriptions and images for each Dot. In the preferred embodiment, Dot Definitions 104 are published to categories specified by the Dot developer. Categories can also be automatically created to support indexing Dots that are most popular (most commonly accessed, most commonly shared, or other sorting criteria).

[00153] Dots as well as their index are formatted as Internet content. Therefore, users can browse or search the Dot index for new Dots using either a browser or using

~~their Home Dot application 200 application (via an AddNewDot Dot or like functionality).~~

~~**[00154]** Each of these Dot templates (Dots definitions 104) are pointed to by a Dot link 321; the Dot link 321 for each Dot is generated automatically by the server 153 (when the Dot is published or previewed) to reference that Dot. Clicking this Dot link 321 adds the Dots definition 104 to the user's Home Dot application 200 (via the user profile 310) and turns the Dot on (instantiates it and fills it in with internet Dot content).~~

~~**[00155]** Once a user receives a new Dot (Dots definition 104) by clicking on a Dot link 321, the Dot template is downloaded and becomes a Dot Definition 104 forming part of their user profile 310 (the XML data that defines which Dots the user has, which Dots they left open, what the state of the Home Dot application 200 last was, etc.).~~

~~**[00156]** The Dot server 153 may optionally host Dot Definitions 104 as part of user profiles 310. Every Home Dot application 200 user has a user profile (expressed in XML) that stores the Dot Definitions 104 that the user has collected as well as the last state of the Home Dot application 200. (See also the specification of the <ALL-CONFIG> DTD in Appendix B.)~~

~~**[00157]** In an additional optional embodiment, when a user successfully logs into the Home Dot application 200, state restoration may be provided by Dot server 153. In this embodiment, authentication may be handled by the Dot application server 153 (as~~

opposed to the user device and Home Dot). The Home Dot application 200 requests the user profile's <ALL-CONFIG> file 311 by communicating with Dot server 153 at 303. The Home Dot application 200 then receives the <ALL-CONFIG> file 311 from Dot server 153 at 303, and restores its state (from the <ALL-CONFIG> 311) presenting last user state, i.e., which Dots were ON, where the Home Dot application 200 was positioned on the screen, etc.)

[00158] The Home Dot application user interface (UI) enables a user to turn a Dot ON 324; in this case the Home Dot application 200 has the Dot Definition 104 (part of the user profile 310 that was retrieved on login) that contains enough initialization data to instantiate the Dot and fill it with content just as the user left it.

[00159] As previously mentioned, Dot Definitions 104 that are part of a user's profile 310 may differ with use. In other words, the Dot Template from which they were originally created may have a different state than a Dot Definition 104 that has been modified through use. The user profile could also be implemented to point to Dot Definitions 104 that are always hosted remotely (and/or not entirely part of the user profile).

[00160] Centralization of Dots within the Home Dot 200 enables users to collect Dots. This feature is significantly different than today's model of the Internet in which users visit a page one at a time, and then leave. Users collect Dots, keep them running, and share them with others. By packaging Internet content as a Dot and referencing it by

a Dot link 321, Internet content is given viral characteristics (i.e., Dots can be instantly distributed). (See also the Session_Config Example found in Appendix B.)

Shares

[00161] Dot servers 153 also host Dot Definitions 104 as part of Shares. Because a Dot link 321 represents a Dot, this Dot link 321 (and hence the Dot) can be easily shared (distributed via email) from one user to other users.

[00162] If a user receives a Home Dot application share and has the Home Dot application 200 installed (their machine is Dot enabled) then clicking on the Share Link Dot link 320 in the share will add the Dot(s) in the share to the user's Home Dot application 200. If a recipient of a Share Link doesn't have the Home Dot application 200 installed, then the Home Dot application 200 is downloaded and installed (with the user's cooperation) with the first Dot(s).

[00163] When a user shares Dots, their Home Dot application 200 generates a share file (XML file) that contains the Dot Definitions 104 of the Dot(s) included in that share. The Share XML is then sent to Dot server 153; the Dot server 153 automatically generates the Share Link 320 that references the Share XML. This Share Link 320 rather than the Share XML is sent or distributed (via email or posted on a web site) to other users (See also the specification of the <SHARE> DTD found in Appendix B.)

~~**SOFTWARE PRODUCT DOWNLOAD PROCESS TO CLIENT**~~

~~**[00164]** Reference is now made to Fig. 11, which is a flow chart 400 illustrating the software product client download process. According to one embodiment, to begin, a user makes a request for a Dot at step 402. At step 402 it is determined that the user does not have the Home Dot application 200 installed. Step 406 comprises of the following:~~

~~**[00165]** User is informed of need to download software to view Dot. The message could be from the site owner (content provider) or from a Dot server.~~

~~**[00166]** "Do you trust [provider/Dot server host]?" dialog is presented to user at 405. A dialog then tells the user the size of download and approximate time for download.~~

~~**[00167]** In step 408, the user clicks the "Yes" button. (Alternatively, if the user clicks the "No" button, step 409 executes with an exit message such as "You decided not to get the Dot"). Proceeding now to step 410, the user gets a Trust Certificate (a security process well understood in the art). The user accepts the Trust Certificate in step 412. (Alternatively, if the user declines, step 413 executes with an exit message such as "You decided not to get the Dot"). In step 414, the user accepts and receives the license agreement and is presented with a dialog that asks the user to confirm the directory for download location. (Alternatively, if the user declines, step 417 executes with an exit message such as "You decided not to get the Dot"). In step 416, the user accepts the~~

license and confirms the file location, and proceeds now to step 418. The download process begins, followed by the installer download and background install operation. The requested Dot opens, and the "KeepMe" Dot opens. Note that the "KeepMe" Dot may be provided by a third party such as a content provider. If no third party-provided "KeepMe" Dot exists, a default "KeepMe" will open.

[00168] In step 420, if the user decides to keep the Dot, the KeepMe (Dot) Operation initiates (see below). If the user decides to close the requested Dot before selecting "KeepMe" and then decides to select "KeepMe" the requested Dot will close, the Keep Dot operation will initiate and the requested Dot will be added to the Dot list. Alternatively, if the user closes the requested Dot and the "KeepMe" Dot without selecting "KeepMe" the Home Dot application 200 will remain installed but the Home Dot application 200 UI will not be available to the user. Lastly, if the user decides to close the "KeepMe" Dot before closing the requested Dot the user will not be able to initiate the Keep Dot operation.

[00169] The other branches of the flow chart 400 will now be described. Beginning once again with step 402, the user requests Dot. According to the alternate branch leaving step 404, the Home Dot application 200 is found on user's machine. In step 405, it is indicated that the user is not registered. (Alternatively, if the user is registered, the Add Dot operation is executed step 407 and the process terminates). Proceeding therefore to step 411, the requested Dot opens. The "KeepMe" Dot opens ("KeepMe" Dot may be a default Dot or a "KeepMe" Dot provided by the Dot/content provider). If the user decides to keep the Dot, the Keep Dot operation then initiates (see

below). Alternatively, if the user decides to close the requested Dot before selecting "KeepMe" and then decides to select "KeepMe" the requested Dot will close, the Keep Dot operation will initiate and the requested Dot will be added to the Dot list. In another scenario, if the user closes the requested Dot and the "KeepMe" Dot without selecting "KeepMe" the Home Dot will remain installed but the Home Dot UI will not be available to the user. If the user decides to close the "KeepMe" Dot before closing the requested Dot the user will not be able to initiate the "KeepMe" operation.

[00170] The following is a summary of the "KeepMe" registration operation (assuming an unregistered user has a partner's Dot open):

- User clicks "KeepMe" button of "KeepMe" Dot.
- User registration form launches.
- User form displays in "KeepMe" Dot.
- User form includes link to privacy statement.
- User completes user registration form and submits.
- Home Dot appears.
- Welcome Dot appears.

[00171] In one embodiment of the present invention, it is possible to track the referring partner, for example for awarding incentives for referring a user. The form of the incentive is a matter of business choice. However, in such an embodiment, the registration operation will comprise the additional step of:

~~Crediting the referring party (e.g., partner) with the referral and/or converting user.~~

~~**[00172]** For an alternative software download process, refer to U.S. Provisional Patent Application ser. no. 60/176,687, Appendix F - DoDots Feature Priority List PPA.~~

~~**[00173]** It will be appreciated that the present invention addresses scaling issues by breaking content up into smaller, more focused software components. These smaller software components (Dot Definitions 104) may thus be served to mobile devices to compensate for bandwidth and content feature support issues, to overcome a lack of browser functionality and processing overhead, etc. Furthermore, the architecture supporting Application Media Packages or Dots offer a consistent experience with application media or Dot content 104 across device types, e.g., similar experiences as between desktop, laptop, web-enabled phone, PDA, etc. As Dots are distributed as easily as web pages are viewed/visited, Dots can be instantly distributed and users can stay connected with content providers of their choice without having to download custom client software from each provider, and without regard to the device type being operated.~~

~~**[00174]** For further details on the structure of the XML structure used for Dots according to one embodiment of the present invention, see Appendix A hereof. For further details on the functions and design of Dot server 153, see Appendix B hereof.~~

~~**APPLICATION MEDIA VIEWER - THE HOME DOT**~~

~~**[00175]** One unique aspect of the present invention is the interaction between the application media package 104 and the application media viewer 119. This aspect is now discussed.~~

~~**[00176]** As mentioned, the application media viewer, or Home Dot, is a network enabled client application. With respect to the user experience, the Home Dot provides the parsing and rendering function of the application media packages or Dots. It also provides for the application behavior of Dots by executing calls or methods that are parsed from the Dot by the Dot Definition, user events, system events, or the like. That is, at a minimum, a Dot comprises a definition of a graphical user interface (frame) and content to be rendered within or associated with that frame. Thus, a critical function of the Home Dot is to parse the Dot definition, render the frame, obtain the content, and render the content in or associated with the frame.~~

~~**[00177]** As previously discussed, the Home Dot comprises Internet content. Thus, the Home Dot is in part a content parser, providing rendering of the graphical user interface (GUI) from Internet content.~~

~~**[00178]** Furthermore, Dots originate on remote network devices. Thus, another function of the Home Dot (client-side) is to obtain and organize Dots on the computer on which the Home Dot resides.~~

~~**[00179]** The Home Dot is a part of a client/server system. As a client installed in one of a multitude of client computers, all communicating with a Dot server, the Home Dot maintains communication with that server to perform system functions for the client. These system functions require a defined command set or functions calls between the client and server. Such functions may include user login, user account status, use statistics, Dot downloads, individual Dot configurations or user customizations, Dot organization, revision updates for both Dot and Home Dot installations.~~

~~**[00180]** Because the present invention enables versatility in the manner of packaging and operating with Internet media (forming application media packages), and because the breadth of internet media and the scope of available information is expansive, the present invention also provides for methods of Dot organization on a user's client computer. More specifically, the Home Dot provides for Dot management for a large number of Dots. Such methods of management include grouping Dots into an organized layout for persistent display, categorizing and grouping Dots into Dot Packs, opening and closing sets of Dots or Dot Packs according to a user's current information requirements, and configuring individual Dots either by user defined categorization or customization parameters that have been enabled by a Dot's developer.~~

~~**[00181]** The Home Dot according to the present invention provides for methods for grouping Dots, sharing information between Dots, and sharing these groupings with others. In other words, the present invention provides for methods by which a user~~

~~or other third party or collectively, third parties, may build upon the utility of Dots as an atomic media element by adding their context through grouping and configuration. In doing so, the present invention enables the addition of third party knowledge that may or may not include the participation of the original Dot developers. Furthermore, the present invention provides for mechanisms by which this knowledge is shared. These mechanisms include saving the customization states of individual Dots, grouping of a plurality of Dots and saving them to a Dot server, and emailing links to other users or installing these links in a web page. Additionally, the present invention provides for third parties to share collaborative efforts by posting such groupings and customizations on a Dot server for general availability.~~

~~**[00182]** In addition to grouping and customization, sharing information and events between Dots is provided for by the present invention. The Home Dot routes messages between Dots and the system. Messaging enables Dots to act upon one another such that an action or event that affects one Dot may also be routed to another Dot, thus affecting a change to it or an action to be taken by it. The resulting actions or changes of the individual Dots may not be the same. To support messaging generally, levels of restrictions are also provided for. It is therefore possible to control the scope of messages received by a Dot from other Dots according to membership or domains. Likewise, it is possible to control transmission of messages to other Dots. Therefore, the present invention provides for cooperative messaging and information sharing between Dots as enabled by Dot developers.~~

[00183] In one embodiment, the server is a high availability system comprising a plurality of individual servers functioning together on a server network. Such a configuration advantageously services a large number of simultaneously executing client Home Dots. Whereas the client application or Home Dot provides for those services surrounding enabled or instantiated Dots, the server network and its applications as embodied in a web site, Java servlets, an RDBMS, Dot and Home Dot distribution support, provides for the development and distribution of Dots. Additionally, as the Home Dot performs the optional features of collecting statistics surrounding Dot usage and posting these statistics to the server, the server may aggregate these statistics for reporting. According to one aspect of the invention, the server operating together with the Home Dot, may source or enable messages to specific Dots based upon real-time context of user interaction with Dots. This function enables, for example, real-time context based advertising. (See also the Client-Server Protocols section of Appendix B.)

METHOD OF DELIVERING DYNAMIC WEB DATA WITHOUT WEB BROWSER

[00184] With reference again to Fig. 6, according to one embodiment of the present invention, Home Dot application 200 enables users to collect Dots 120, 230, etc., organize those Dots into collections, manage the collections with organizational and workspace management functionality (e.g., categories, Snapshots, group-move), and easily share Dots, Dotpacks, and Snapshots with others. Also, the Home Dot application 200 implements and enables a Dot Messaging Architecture (DMA,

discussed further below) providing messaging between Dots and the Dot server system.

[00185] When Dots are instantiated on a client machine, the Home Dot application 200 collects usage statistics (211, 212, 213, 214, 215) and keeps an event log (in XML format) that is intermittently sent to Dot application servers 153 and stored, for example as part of the Dot database 202 as use statistics 206. This enables the Dot server to track how the Home Dot application 200 and the Dots are being used and shared. In one embodiment this is implemented as a local MFC (Microsoft Foundation Class) application on the Windows platform, thus enabling rapid user response (no Internet delay for functional UI components). Home Dot application 200 functionality may also be provided across platforms (MAC O/S, Linux, unix, mobile, wap, etc.)

HOME DOT OPERATION

[00186] Each user of a Home Dot application 200 has a Home Dot application login account 205 that accesses (step 303 of Fig. 7) a personal user profile stored on a Dot server 153 that stores the last state 212 of the Home Dot application 200 along with which Dots a user has collected 213, and how the user has organized their collection according to categories, Snapshots or packs.

[00187] When the Home Dot application 200 is launched, it queries the network for configuration information, enables a user to login, retrieves the user's profile from

~~the Dot application server 153, and restores the Home Dot application 200 to the last state that was stored to the application server 153.~~

~~**[00188]** The method used by Home Dot 200 to access remote configuration information (again, step 303 of Fig. 7) creates a very flexible application that can be configured to support different application looks, different login sequences, dynamically by session and for each for each user. The user profile retrieved at login 303 enables the application to be personalized for a user, and allows that user to access that personalized application state on different client computers or devices 199.~~

~~LAUNCHING THE HOME DOT APPLICATION~~

~~**[00189]** When the Home Dot application 200 is launched, it reads the registry on the client machine to find the location of a configuration file (session-config). The session-config (XML file) directs the Home Dot application 200 to an application server using a URL as the server address that will handle its servlet requests. The URL points the Home Dot application 200 to a default graphic element (skin) and generic Dot graphic elements for controls such as the Title Bar 164 (Fig. 4), Bottom bar 166 (Fig. 4), etc. and it points to Dot definitions 213 for several default Dots, such as a Help Dot, Add New Dot, and Login Dot (i.e., System Dots that support the Home Dot application 200, as described further below).~~

~~**[00190]** For security, the session-config is served by a servlet method operating on the Dot application server 153 that only responds to authorized Home Dot~~

~~applications 200. The session-config can also be served as a flat file from a standard web/local directory.~~

~~**[00191]** The session-config file contains a mechanism to redirect the Home Dot application 200 to another session-config file. This mechanism may look like:~~

~~<REDIRECT> = "URL" attribute:~~

~~When the Home Dot application 200 reads the redirect attribute, it ignores the current configuration (session-config file) and attempts to retrieve the configuration stored at the redirect URL. The Home Dot application 200 will not redirect if this tag is omitted, if the URL is empty, or if the URL is the same URL used to retrieve this configuration in the first place. (This last state is recursive; therefore if redirection were carried out, the two configuration files redirect to each other, resulting in an endless loop within the Home Dot application 200.)~~

~~**[00192]** The session-config file contains a mechanism to force or provide an option to upgrade the Home Dot application. This mechanism employs a version tracking, such as:~~

~~<VERSION> = "string" attribute:~~

~~This attribute is required and must match the version of the Home Dot application 200. According to one embodiment, if this attribute is omitted or differs from the Home Dot application's 200 version, the Home Dot application 200 will download the executable~~

specified in the UPGRADE attribute and execute it. The mechanism for performing the upgrade may look something like the following:

<UPGRADE> = "URL" attribute:

This is used only if the VERSION attribute differs from the Home Dot application's 200 version. This URL specifies the location of an executable that will upgrade the Home Dot application. The Home Dot application 200 will quit, then run the upgrade executable automatically.

SYSTEM DOTS

[00193] According to one embodiment of the present invention three system-level Dots are provided with a Home Dot: the AddNew Dot; the Help Dot; and the Login Dot. Dot definitions for these System Dots are referenced in the session-config. System Dots are Dot definitions that are not explicitly listed in the Home Dot application 200 or a user's ALL-CONFIG, but are accessible via the Home Dot user interface.

[00194] The AddNew Dot enables users to access the Dot Index through the Home Dot application 200 and find new Dots and Snapshots to collect. The Help Dot contains help content for the Home Dot application 200.

[00195] The Login Dot is what first comes up when the Home Dot application 200 is launched. It enables the user to login to the Home Dot application 200.

~~**SAMPLING DOTS -TRIAL DOTS**~~

~~**[00196]** One feature of the present invention is the ability to provide a user with the option to try, or sample Dots. According to one implementation of this feature, if a user doesn't have a Home Dot application login account (or is not logged in), then the Home Dot application 200 is in trial mode. Dots can be sampled (by clicking on Dot links 321) but not kept.~~

~~**KEEPING DOTS**~~

~~**[00197]** A user may decide to keep or not keep a Dot. According to one embodiment, to keep a Dot, a KeepMe Dot is displayed with Dots that are sampled (unique to Dot developer or provider) that informs the user that a Home Dot application 200 login account is required to keep the Dots. The KeepMe Dot has a mechanism (link) that enables users to create a new Home Dot application 200 login account 205 and keep the Dots by adding them to the new user's Dot definitions 213 or user-profile 310 (Fig. 7).~~

~~**USER LOG IN**~~

~~**[00198]** Each Home Dot application user has a Home Dot application login account that accesses a personal user profile (stored server 153). When a user launches and logs in to the Home Dot application 200, it retrieves the user profile from server 153 and restores the Home Dot application 200 to the state in which the user left it, thus~~

recovering their personal application experience. When the user logs out, their user profile is updated on server 153.

[00199] This mechanism enables different users to use the same client computer 199 and receive different application experiences or the same user to use different client computers or devices 199 and receive the same coherent experience. The application 200 restores itself to its last saved state no matter where a user may log in.

[00200] After the Home Dot application 200 retrieves its configuration information, it renders the Login Dot from the Dot definition specified in the SESSION-CONFIG. The Login Dot's Dot content asks the user for login and password, sends the login request to the Dot server servlet (also specified in the SESSION_CONFIG), and retrieves the user's profile required to restore the Home Dot application.

USER PROFILE (<ALL_CONFIG>)

[00201] A Home Dot user's profile holds the state of the user's Home Dot application 200 (size, position on the screen, which Dots were open) along with which Dots 213 the user has collected, and how the user has organized their collection (categories, Snapshots or packs). The profile is stored in an XML file called the user's <ALL_CONFIG>. The ALL_CONFIG file contains a SHARE and a LASTSTATE element as described below. The Share may look something like the following:

ALL_CONFIG's <SHARE> attribute:

~~Since a Share can contain Dots and Snaphsots, the Home Dot 200 uses the SHARE XML DTD to represent the user's collection of Dots and Snapshots. The LASTSTATE element may look something like the following:~~

~~ALL_CONFIG's <LASTSTATE> attribute:~~

~~It contains a PRESET element 313 and a HOMEDOT 314 element (Fig. 7). The Home Dot 200 uses the PRESET XML DTD (Snapshot) to represent the Dot's that are left open. The Home Dot application element holds the position/size/state of the Home Dot itself. (See also the All_Config Example found in Appendix B.)~~

~~THE HOME DOT APPLICATION SERVER~~

~~**[00202]** The Dot application server 153 support much of the functionality of Home Dot application 200 (such as logging in/out, collecting, adding, and sharing Dots and Snapshots, Home Dot usages logging, etc.)~~

~~**[00203]** All communication 210 between Home Dot application 200 and the Dot application server 153 is secure and occurs over standard communication protocols (HTTPS). HTTP/HTTPS is chosen for the socket level client/server communication protocol because of its simplicity and more importantly, because most firewalls typically leave the default handling of the HTTP/HTTPS communication with the client.~~

[00204] The Home Dot 200 communicates to the server 153 via an HTTP request. The HTTP request URL contains an API call. Any API calls that require the uploading of data to the server place the data in the message body of the request. API calls that require uploading/downloading of data send/receive that data in XML format. All API calls are the end part of a complete URL that begins with:

http:ll<someservername>.DoDots.com/DoDots/

where <someservername> is variable and DoDots is an alias for the DoDots servlet/JSP servlet or servlets directory. Arguments to any of the methods are passed in as name value pairs in the query string portion or the URL.

[00205] The preferred embodiment of the Dot application server application that supports the Home Dot application 200 currently supports a number of servelt methods including the following:

AddUser

[00206] The adduser method is used to create a new Dot user account. Note that this is available as an end-user API to allow new users to add themselves rather than wait for an administrator to do it for them.

~~GetUser~~

~~[00207]~~ ~~The getuser method retrieves the user's own demographic data. This method supports the ability of the Home Dot application 200 to then allow updates to the user's demographic information using the setUser method (see below).~~

~~SetUser~~

~~[00208]~~ ~~The setUser method is used to update an existing user account. Note that this is available as an end-user API to allow existing users to update themselves rather than wait for an administrator to do it for them.~~

~~GetSession Config~~

~~[00209]~~ ~~The getsession Config method is used to retrieve general Home Dot application configuration information.~~

~~Set Password~~

~~[00210]~~ ~~The setPassword method provides a mechanism for the user to change their associated Dot server account password.~~

~~getMasterDotList~~

~~[00211]~~ ~~The getMasterDotlist method requests the list of all Dot templates known by the server 153. The server 153 returns a message body in predefined XML format~~

(see the DOT_MASTERLIST DTD and example) of all possible Dots and their default template values. Note that included in each Dot element are a version stamp and dotclass ID's which allows the Home Dot application 200 to determine if it has the most up-to-date Dot template and Dot class binaries for that Dot. The Home Dot application 200 may then request the updated Dot template (see the getDotTemplate method) and/or dotclass binary zip file (see the getDotClassBinaries method).

setAllConfig

[00212] The setAllConfig method sends, at 303, the entire body of user-specific client-side configuration information 310 including all Snapshots, Dots, general settings, etc. to the server 153 for persistent storage on behalf of the user. The user profile data 310 is sent as the message body in a predefined XML format (see the ALL-CONFIG DTD and example). The response returned by the server 153 indicates success or failure.

getAllConfig

[00213] The getAllConfig method retrieves, at 303, the last user-specific, server-side saved client side configuration information 310 (all Snapshots, Dots, general settings, etc) from the server 153. The data is sent as the message body in a predefined XML format (see the ALL-CONFIG DTD and example).

~~SetAl I Events~~

~~[00214]~~ ~~The setAllEvents method sends, at 307, usage information (211, 212, 213, 214, 215, Fig. 6) that has accumulated since the last call to setAllEvents to the server 153 for later use in statistical usage report generation (for DoDots's own use). The data is sent as the message body in a predefined XML format (see the ALL-EVENTS DTD and example). The response returned by the server indicates success or failure.~~

~~GetDotTemplate~~

~~[00215]~~ ~~The getDotTemplate method requests from the server 153 a specific Dot Template (Dot definition) from the Dot database. The server 153 returns the XML that describes the Dot. This method is called when the Home Dot application 200 has determined that it does not have the current version of the Dot and the user wants the most recent version. (NOTE: this includes when the user is downloading the Dot for the first time)~~

~~Addshare~~

~~[00216]~~ ~~The addshare method stores at 302 a specific shared Dot(s) or Snapshot(s). The server 153 takes the XML that was uploaded by the Home Dot application 200 for sharing and returns the unique id of the share. The Home Dot application 200 puts the XML stream to be shared in the request body.~~

GetShare

[00217] The getshare method requests from the server 153 a specific shared Dot(s) or Snapshot(s). The server 153 returns the XML that was stored for sharing by the original sending user. The format of the returned XML stream is dependent on what has been shared.

CACHING LAYER

[00218] Returning to Fig. 6, the Home Dot application 200 may advantageously implement a caching layer 219 between its network requests and the network 210. The Home Dot application 200 sends network requests to the caching layer and the caching layer is responsible for completing those requests (retrying if necessary).

[00219] This is used in the event that the Dot application server 153 does not respond. If a user is attempting to log in and the server 153 does not respond, then the Home Dot application 200 logs in using a locally cached <ALL-CONFIG> (the last written user application state can be saved in the caching layer 219). If a user attempts to log out and the server 153 does not respond, then the caching layer 219 retries at a later time to write the logout application state to the server 153.

ADD NEW DOTS/SNAPSHOTS (PACKS)

[00220] To get a Dot and invoke it - that is, turn it on - the Home Dot application 200 requires the Dot's definition (XML file with the information necessary to instantiate

~~a Dot and point it to Internet content). With reference again to Fig. 7, a user collects Dots from multiple sources: by clicking on Dot links 321 (Internet link to a Dot's Dot definition) and/or by receiving a Share 320 (Dots and Snapshots shared from other users). In both cases, the Dot definitions for the Dots involved are hosted and served by a Dot application server 153.~~

~~[00221]~~ ~~Users can find Dots (Dot links) in the Dot server 153 DotIndex 204 (online Database of Dots via a web browser or the AddNew Dot) and/or promoted on a content provider server 155, e.g., via a web site. When a user clicks on a DotLink 321 and the Home Dot application 200 is installed, the Home Dot application 200 retrieves the Dot definition that the DotLink 321 points to, adds it to the current user's ALL-CONFIG file 311 of user profile 310, and turns the Dot on 324. The Dot instantiates the UI and begins filling it with the Dot content (Internet content) as defined in the Dot definition.~~

~~[00222]~~ ~~Once a user receives a Dot, the Home Dot application 200 saves the Dot's definition as part of the user's ALL-CONFIG file 311. Because a Dot definition becomes part of a user profile 310, it can be modified by use e.g., its default TitleBar image 132 may be changed or its WebConduit control may be navigated to a different URL using DMA messages. The Dot will pick up where it left off next time it is turned on.~~

~~[00223]~~ ~~Dots can also be received in the form of Shares. A Share 320 is a XML file that represents a collection of one or more Dots and/or one or more Snapshots. A~~

Share 320 may be shared, for example by attaching a Share Link 320 to a standard email message. When a user receives an emailed Share Link 320 from another user and clicks on the Share Link 320 in the email:

- The Home Dot application 200 retrieves the share XML file from the Dot application server 153 using the getshare servlet method to which the Share Link 320 points.

- The Home Dot application 200 parses the share package and adds the new Dots and Snapshots to the Share recipient's ALL-CONFIG 311. The Share recipient can now turn on any Dot and Snapshot received as part of the share.

[00224] According to one embodiment of the present invention, every Dot that a user gets is added to and becomes part of the user's ALL-CONFIG 311 and becomes accessible to the user anywhere they can login to a Home Dot application 200. According to another embodiment, a user is provided with the option to keep or discard a sampled Dot prior to adding it the user's ALL-CONFIG 311.

[00225] If the Home Dot application 200 is not installed on a client computer 199 when a user first attempts to get/receive Dots, then the Home Dot application 200 is downloaded and installed with the first Dots that are received. From that point forward, the client machine 199 is Dot-enabled and the user is not required to download and install the Home Dot 200 again (subject to updates). This method removes the

alternative approach known today in the art which is to download a custom client application from each company's web server 105 that is visited.

[00226] Once a client machine 199 is Dot-enabled, (i.e. the Home Dot application 200 technology installed), then every time that a user clicks on a Dot link 321, the Dot 120 pops up immediately (faster than it would take to load a small web page). Rather than download and install software for different Dots, only the Dot definitions 102 which are packaged web readable content is collected.

DOT CATEGORIES

[00227] The Home Dot application 200 enables users to organize Dots that they have collected (in their ALL-CONFIG 311) by categories. A Dot can be a member of more than one category. Alternatively, a Dot is not required to be categorized (i.e., set attribute categorized = uncategorized). The Home Dot application 200 enables the user to Add/Remove/Rename categories and edit a category's contents (those Dots that are members of that category). Categories may also be assigned by the Dot creator (which may or may not be overwritten.)

[00228] In the preferred embodiment, Dots keep track of the categories to which they are members (e.g., there is no master category list). The categories to which a Dot belongs are added to the Dot's definition (in the user's ALL-CONFIG 311).

[00229] The Home Dot application 200 has one unique category: the ALL MY DOTS category (users cannot rename or remove this category). This category contains

all the Dots that the user currently has in their collection (in their ALL-CONFIG 311). Removing Dots from the ALL MY DOTS category removes it from ALL-CONFIG 311. A user can duplicate a Dot from within the ALL MY DOTS category in the event they want two of the same Dot (e.g., 2 stock watching Dots – one for monitoring a first security, and the other for tracking a second security).

MANAGING DOTS

[00230] The Home Dot application 200 provides for several features enabling visually organizing, collecting, and working with Dots 102 individually and as a system. Referring to Figs 12A, 12B, and 12C, a series of drawings depicting a user interface 600 for the organization of Dots on a computer desktop is shown.

[00231] To improve ease of use when working with Dots in groups, the Home Dot application 200 enables users to capture an image, or Snapshot, of a workspace that they have defined with Dots. Such Snapshots may include, for example, which Dots are on and where they are positioned on a display screen 656. Users can instantly recall

these Snapshots to restore the previously defined set of Dots to their captured locations on screen 656.

[00232] Snapshots can be created to support different work tasks (e.g., morning news pack, web development pack), different user groups (e.g., small business pack, runners health pack, school study pack), or different activities (e.g., day trading,

dream team sports tracking). Within a Snapshot may be user-created Share Packs 659 (groupings) of Dots (created, for example, using Share Links 320). In addition, Dot developers can create and publish Packs, which are groups of Dots designed to work together or which may be of similar/related interested to a user, and online content providers can offer their users pre-made Packs.

[00233] Users can easily and quickly create and configure/reconfigure their own aggregation of content and functionality as Dots, thus providing a user-aggregated content/application system. This is significantly different than current Internet functions of offering users "My-" versions of a full-screen Internet site in which the choice of content, arrangement, presentation style, etc. are quite limited.

[00234] Dots may be aggregated together and manipulated as a block. Blocks are defined as linear clusters of Dots (snapped together vertically or horizontally). An example of a group 659 of Dots 647, 649, 651 not in a block is shown in Fig. 12A. That is, in Fig. 12A the group 659 of Dots are not aligned with one another with reference to either a horizontal or vertical axis. Two examples of the Dots 647, 649, 651 comprising group 659 arranged in a blocks are shown in Figs. 12B and 12C, respectively.

[00235] Blocks of Dots have an orientation: vertical (Dots snapped to each other in a vertical column as in Fig. 12B) or horizontal (Dots snapped to each other in a horizontal row as in Fig. 12C). In one embodiment, the Home Dot application 200 or Dots (647, 649, or 651) expose a UI mechanism (e.g., CTRL+O, see slamming) to enable users to change the orientation of Blocks of Dots.

~~**[00236]** Blocks of Dots can have a justification. When a block of Dots are "justified", they share a common edge such that all edges on one side of the Dots are lined up. For example, a vertical block of Dots all snapped toward the right side of the screen 653 is right justified as shown in Fig. 12B. In one possible embodiment, the Home Dot application 200 or the Dots exposes a UI mechanism (e.g., SHIFT+CTRL, CTRL+J, see slamming) to enable users to change the justification of Blocks of Dots (left, center, right).~~

~~**[00237]** Slamming blocks of Dots against screen edges (653, 655) or other window edges is used as a mechanism to alter the justification and/or orientation of a Block of Dots. For example, a user may slam a horizontal block against the left edge of the screen and the Block's orientation could swing vertical and justified to the left with all Dots snapped to the left of the screen. Likewise, a horizontal block that is bottom justified may be slammed against the top edge 655 of the screen changing it to a top justified Block.~~

~~**[00238]** One feature provided by an embodiment of the present invention to assist a user with the organization and presentation of Dots is the snap feature. A Dot (649 for example) "snaps" to other Dots (647 and 651 for example), other windows (not shown), and screen boundaries (edges) 653, 655. When Dots 120 are dragged (moved) near an edge (653 or 655), a magnetism behavior is exhibited and the Dot 649 accelerates towards and "snaps" to that edge (653 or 655). The present invention provides for a magnetic gap 658 such that even when Dots are snapped to edges, there still exists a~~

gap (~5 pixels). This gap is supported so that the user may easily visually recognize independent Dots.

[00239] The snapping feature simplifies the task for users to quickly and neatly align Dots. (See Figs. 12B and 12C). When Dots (647, 649, and 651) are snapped together and a Dot is collapsed or resized, then a Dot that is snapped to another Dot can move accordingly; for example, staying snapped or not, according to user preference.

[00240] The user interface of the present invention supports the movement of Dots (647, 649, or 651) in clusters. All Dots (647, 649, or 651) that share an edge (snapped together = cluster/group) can be grabbed and moved as a group 659 (which may, but need not be, a block).

[00241] The present invention teaches multiple methods of selecting a cluster of Dots 659 (as opposed to a single Dot 651 for example). The simplest method comprises of pressing the CTRL key before grabbing a member Dot (any one of 647, 649, or 651) of the cluster 659. This method selects the entire cluster and moves all Dots 120 as one unit.

[00242] Another way is to change the user's selection, that is which Dots (647, 649, or 651) in the cluster 659 are selected for moving, by the number of times a user clicks before grabbing a member Dot (647, 649, or 651) of the cluster 659. For example:

- 0 clicks before grabbing a member Dot (647, 649, or 651) selects and moves the entire cluster 659 of Dots.

- 1 click before grabbing a member Dot (647, 649, or 651) selects and moves just the member Dot.

- Additional clicks could select all vertical, all horizontal, etc.

[00243] The present invention provides for a unique method of indicating which Dots (647, 649, 651) in a cluster 659 are selected (for a group action such as move, or minimize). When multiple Dots are selected (by either method: CTRL+Select or Click+Select), a halo 661 appears around the selected Dots. In one embodiment halo 661 (shown only in Fig. 12A) is a contrastingly shaded or brightly colored line (~3 pixels wide) floating around the outer perimeter edges of the Dots selected (647, 649, 651) in a cluster 659. The halo remains visible as the Dots (647, 649, 651) are moved.

DOT SHARING

[00244] The Home Dot application 200 provides methods for users to easily and quickly Share the Dots that they have collected and the Snapshots that they have created with others. An exemplary delivery mechanism is email. The user's default mail client application is used (a user can use their existing address book and add a detailed message); a web-based mail service extended through the Home Dot application 200 or Dots may also be used. To share and convey shared Dots or Snapshots in an email, a Share Link 320 is required (an Internet link that points to stored Share data on the Dot application server 153).

~~**[00245]** When a user sends a Share or Snapshots to another user:~~

~~1) A Share XML file 302 is constructed employing the shared Dots and Snapshots from the sharing user's ALL-CONFIG 311.~~

~~2) The Share XML file 302 is stored in Dot database 202 on Dot application server 153 in exchange for a unique shareID (using the addshare servlet method).~~

~~3) An email is constructed with a Sharelink 301 (an Internet link that points to the stored Share) and placed in a new email using the user's default mail client.~~

~~**[00246]** When a user receives a Share of Dots/Snapshots from another user and clicks on the ShareLink 320 in the email:~~

~~1) The recipient's Home Dot application 200 retrieves the Share XML file from the Dot application server 153 (using the getshare servlet method) to which the Share Link 320 points.~~

~~2) The Home Dot application 200 parses the Share package and adds the new Dots and Snapshots to the Share recipient's ALL-CONFIG 311.~~

~~**[00247]** The Share recipient can now turn on any Dot and Snapshot received as part of the Share. Because the Share was constructed from Dot definitions that were~~

part of the Sharing user's ALL-CONFIG 311, the recipient receives the Dots just as the Sharing user had configured them at the time of building the Share.

PREFERRED EMBODIMENT OF CLIENT / SERVER SYSTEM

[00248] The client/server model 500 of the preferred embodiment is described to according to Fig. 13. Paramount for the successful implementation of a consumer or high volume system is the proper separation of responsibilities between Dot clients (501, 502, 503) and the Dot server system 555. The architecture presented herein supports the primary operations of Dot distribution, client installation, use statistics collection, as well as Dot developer activities. These operations, as described above, are supported in part by the XML structure and calls as listed in the attached appendices A and B. It will be appreciated by one skilled in the art that the list is not exhaustive but descriptive of an implementation of certain features of the present invention.

[00249] The Dot server system 555 which is required for support of the general Dot functionality provides for persistent storage and retrieval of configuration and statistical (usage) information and for intelligent software upgrade service for the Dot clients 501, 502, 503. Users are required to logon to the Dot server system 555 with a username and password to access this information. The server 555 also provides for generating statistical analysis reports. The server 555 is used to provide small, persistent data storage areas for third-party Dots. The server 555 may be maintained by a single administration entity and does not require third-party Dot providers for its

support. The content contained by the Dot however, may be provided by the third party developers, with the initial content optionally being supplied by existing or re-purposed web pages served by the third party web servers. An administrative interface is therefore provided according to the present invention which is used by the administration entity to maintain the software upgrade information, manage users and generate statistical usage reports.

[00250] The software portion of Dot server system 555 consists of several architectural components, including:

[00251] Web servers (505, 507, 509), Servlet JSP Engines (506, 508, 510) which maintain a Java based XML Parser with SAX (Simple API for XML) and DOM (Document Object Model) interfaces, all of which share a common file system, and at least one Relational Database Management System (RDBMS) Server 513 which supports the RDBMS file system 514.

[00252] According to one embodiment of the present invention, there are no specific operating systems, application server, or database server constraints placed on the server 555 (e.g. the operating system (O/S) may be NT, Solaris, HP-UX, Linux or FreeBSD, or any other viable server O/S. The RDBMS may be Oracle, Sybase, Informix, SQLServer etc.).

[00253] The client and server transmit and receive data on the internet 504 in XML format over standard HTTP/HTTPS. An XML parser residing on the Servlet JSP

~~Engine (506, 508, 510) assists in deconstructing and reconstructing the XML into and out of the RDBMS Server 513 when the XML stream contains information that is needed for report generation. XML parsers with DOM and SAX interfaces are freely available for most major programming languages. HTTP/HTTPS, and therefore a web server (505, 507, 509), is chosen for the socket level client/server communication protocol because of its simplicity and more importantly, because most firewalls typically leave the default TCP ports for HTTP/HTTPS (ports 80 and 443 respectively) unblocked.~~

~~**[00254]** The Web servers (505, 507, 509) provide the default handling of the HTTP/HTTPS communication with the Dot client 501, 502, 503. The Servlet JSP Engine (506, 508, 510) functions as an in-process extension of the Web servers (505, 507, 509) and provides the infrastructure for the application logic layer (servlets) and the presentation layer (Java Server pages). Servlet/JSP is chosen over the architecturally similar ASP due to consideration in performance, maintenance costs, and the variety of O/S and application server vendor choices.~~

~~**[00255]** The persistent storage mechanism for everything except the binary software components (client component updates) is Relational Database Management System (RDBMS) 513. The binary software component for client updates are stored in the file system 514 with a pointer (full pathname) which is stored in the database to each component file. The binary software components are not stored directly in the RDBMS in so as to improve performance.~~

[00256] Since the configuration data need not be manipulated or reported on by the server, the configuration data is stored directly in the RDBMS 513 as a small text file, in the form by which it is transmitted from the client 501, 502, 503. The configuration data is not deconstructed and reconstructed into its constituent parts by the XML parser on the servers (505, 507, 509). The statistical data is used to generate reports on the server-side by a servlet JSP-based administrative interface. SQL queries are used for statistical report generation to provide simplicity and flexibility. Therefore, a single statistical data XML stream is deconstructed upon receipt by an XML parser residing on the Servlet JSP Engine (506, 508, 510) and stored as discrete RDBMS columns (the entire stream instance is the RDBMS row) rather than storing the entire stream in a single text column.

DEVELOPMENT AND TESTING OF DOTS

[00257] It is possible to provide for the development and testing of new Dots, Categories, Packs, etc. in a physically separate system that is a superset (duplicate with additional testing support) of the production system 500. The development and testing (dev/test) system has additional server methods and interfaces, not present in the production system 500 which requires higher security and simplified maintenance, to facilitate the development and testing process. The interfaces to these additional server methods are HTML/JSP pages to allow for easy accommodation of new, geographically distributed Dot content providers.

~~**[00258]** The development and test system allows the Dot content providers to easily build and test new Dots and Dot Packs. When tested to their satisfaction, the Dot content providers can then, through the interface, submit their Dots/packs for inclusion in the production system. The Dot administrator will be able to periodically run a report to see which Dots/packs were submitted. The Dot administrator is then able to decide whether or not to approve a Dot/pack for import into the production system. Migration of an approved Dot/pack will involve exporting the appropriate data from the RDBMS on the development and test system, moving the export file from the development and test system and importing the export fie into the production system.~~

~~**DOT MESSAGING ARCHITECTURE (DMA)**~~

~~**[00259]** Dots and the Home Dot application have a messaging architecture (Dot Messaging Architecture - DMA) that enables elements of the system including Dots, controls within Dots, and the Home Dot application, to communicate with one another. This enables these elements to exchange information, request actions or functionality, and respond to system, element, or content events.~~

~~**[00260]** Fig. 14 illustrates the message routing paths and elements 700 of the Dot Messaging Architecture. The Dot Messaging Architecture (DMA) has a messaging addressing and routing scheme, defined messages, and an extensible message format (as defined in further detail below) that provides each component with access to component, application, and system features and true application behavior. Also, Dot~~

~~content has access to the application rendering system within the Home Dot 200 and other Dots (711, 751, 761) via the DMA.~~

~~**[00261]** All elements of the system can send and receive (via paths 701, 702, and 703) DMA messages. In this embodiment, these elements fall into three groups: the Dots (711, 751, 761); the Controls in the Dot's control space including the WebConduit control (714, 754, 764) and base controls that encompass menu controls and window operations controls; and the system control 780 embodied within the Home Dot application 200.~~

~~**[00262]** The DMA enables controls to send and receive messages. Controls can exchange messages with other elements in the same Dot (controls in the control space, Dot frame, System) or in a different Dot (the addressing scheme supports addressing elements in other Dots. All Dot controls share a set of common messages. They also can provide messages that are unique to that control (see WebConduit Control). Common control messages include messages such as:~~

~~#show~~

~~#hide~~

~~#get width~~

~~#get height #get size~~

~~#is open~~

#get address

These messages query/effect properties/methods of a control within its Dot's control space such as layout, size, etc.

[00263] The Web Conduit control (714, 754, 764) supports additional messages (in addition to the common control messages explained previously). These unique Web Conduit messages enable other elements in the Dot System (Controls, Dot frames, Home Dot application 200) to interact with WebConduit functionality (request actions or functions, and respond to events). Messages unique to the WebConduit control which is a wrapped Microsoft IE web control in the preferred embodiment, include:

##<any javascript>

#navigate.

[00264] One of the most significant features of the WebConduit control is that DMA enables messages to flow IN and OUT of standard HTML rendered within a control. The preferred embodiment currently extends DHTML and javascript to send and respond to DMA messages. The present invention supports SendMessage (a synchronous method which is used if return result is required) and PostMessage (an asynchronous method in which no return result required) methods that can be called using a window.external.<method> call from DHTML in the Web Conduit control (714, 754, 764).

~~**[00265]** The preferred embodiment of the message format (explained in more detail in the following sections) allows for two types of messages to be sent IN and OUT of HTML with respect to the WebConduit control. Specific messages and functions can be called (e.g., #navigate) that cause a WebConduit to perform a specific action or function. In addition, any javascript (e.g., ##<any javascript>) can be called within a WebConduit's document as well.~~

~~**[00266]** This provides the messaging architecture with exceptional flexibility and extensibility in which data can be passed, functions can be called, and variables can be set. Integrating DMA with standard Web content by enabling DMA messages to call into HTML and for DMA messages to be initiated from HTML within the WebConduit control is the basis that enables Dots to exhibit true application behavior.~~

~~**[00267]** Any DMA message that is defined can be sent or called from DHTML. This, in conjunction with the application logic capability that is provided by javascript (and other HTML scripting languages), provides the application media development environment (which includes application development platform and language). A Dot developer can thus author a new Dot application by developing web content (HTML, GIF files, etc.) and by packaging that content in a Dot Definition, eliminating the need for compilers and consequently, downloading executables. Therefore, if a client device 199 has been Dot-enabled (the Home Dot application 200 is installed) then that device can instantaneously view, open, run Dots as well as modify and save their configurations.~~

[00268] DMA messages can be sent to and received from a Dot. This enables elements of the system that can send and receive DMA messages to interact with a Dot. Elements of a Dot's definition can be accessed/modified using Dot messages. Examples of Dot Properties / Dot methods that can be accessed and modified via the DMA include:

#set-title

#get-title

#set-size <width> <height>

#is-open #close

#collapse #uncollapse

#set-title-images <up-URL> <down-url> <over-url> <inactive-url>

[00269] Since a message can originate from DHTML content which is rendered in the Web Conduit control as Dot content, standard web content when rendered in a Dot can therefore access properties and behaviors of its packaging (the Dot). Internet content can therefore do such things as modify the size of the Dot that is rendering it. It can move it or collapse it.

[00270] The present invention therefore enables properties and behaviors to be tied to and between any Dot content event, even outside of a Dot affecting another Dot (e.g., an internet content (DHTML) mouse-over event can change the size and position of another Dot, etc.)

[00271] Most of the DMA examples thus presented are of other elements sending messages to the Dot frame (711 for example). An example of a Dot frame 711 sending messages can be seen in the case of the menu control 712 (discussed further below). The Dot frame 711 may also send messages (as can the menu control 712) based upon the occurrence of certain events such as Dot-moving, Dot-collapsed/expanded, Dot-infocus/inactive, etc.

[00272] The preferred embodiment has the menu control 712 implemented as part of the Dot Frame 711. Menu control 712 entries are given/tied to actions using the DMA. Each menu entry may consist of, inter alia, a text-title, an icon, a tooltip, an ID, and an action. The action is simply a DMA message that has a recipient specified address. Dot menu control 712 items are defined as part of a Dot definition.

[00273] When a menu entry within a menu control 712 is selected by a user, the specified Dot Message is sent to the specified recipient (DMA address). This illustrates an example of the true application behavior that is exhibited by Dot Content having access to DMA.

[00274] As with the functionality described in the previous sections, the following are examples of types of additional application behaviors that are possible:

- A menu control 712 entry (e.g., refresh) can send a message (to refresh) to a Web Conduit control and cause an action (e.g., refresh content).

~~A menu control 712 entry can send a message to the Dot (itself) and cause it to collapse, or resize, or exit.~~

~~A menu control 712 entry can call any javascript (##<any javascript>) in the HTML of Dot content rendered in the same Dot or a different Dot to set variables, change images, call functions, etc.~~

[00275] ~~DMA messages can also be sent to and received (701, 702, 703) from the system (720) - the functional layer above and between individual Dots. Some aspects for which the system is responsible include DMA message routing, adding, removing, opening, closing Dots, etc. In the preferred embodiment, the system includes the Home Dot application 200 or equivalent Dot-rendering and Dot-management client application.~~

[00276] ~~Examples of system functions and behaviors that can be accessed and called using DMA messages include:~~

~~#refresh~~

~~#install-dot <dot-url>~~

~~#have-dot <dot-address> #delete-dot <dot-address> #quit~~

~~#save-to-server~~

~~#get-screen-width~~

#get-screen-height

#close-all-dots

#open-preset <preset-name>

#take-preset <preset-name>

#get-dot-ids <dot-address>

[00277] Messages can be sent to/from controls (WebConduit control - in and out of HTML), to and from the Dot (DotFrame - menu control 712, Titlebar 713, bottom bar 717, flexible Dot Definition), to and from the system (Home Dot application 200 - open and close Dots, system variables and data). The DMA addressing scheme also provides for messages to be sent to any Dot, to any control in any Dot, to any HTML, in any control, in any Dot. This enables Dots to work together as an application system, enables Dot developers to share functionality and leverage and build on the functionality of other Dots.

[00278] In the preferred embodiment, a DMA message has two components: a recipient address and the message body itself. Both are represented as strings. The addressing scheme is explained in the next section.

[00279] The body of a DMA messages is, at its simplest, a text string which may represent any javascript, for example, sent to a Web Conduit control. For defined messages, there is a method element to define the function/behavior call/request [e.g., #set-size] followed by arguments if the method element requires them (e.g., height in

DotUnits; width in DotUnits).

[00280] The present invention provides for two functions that enable messages to be generated from within DHTML in the WebConduit control: SendMessage and PostMessage. SendMessage, which is synchronous, is used if a return value is required. PostMessage, which is asynchronous, is used otherwise.

[00281] Below are examples of messages generated with DHTML within the WebConduit control:

window.external.PostMessage ("#.:", "#set-position 350 500")

This message is sent to a Dot and causes it to move to a new screen position, for example 350x500 pixels.

window.external.SendMessage ("#system, "#get-screen-width")

This message asks the system for the current screen's width in pixels.

[00282] Messages are routed according to three pieces of information: Domain, Dot Specifier, Control Specifier.

[00283] One of the ways that a Dot is identified is by a Domain. In the preferred embodiment, the Domain is the same as a Dot provider's ID (developer ID). Generally this is a unique identifier for each company or developer and is specified in a Dot's definition.

[00284] Dots in the same domain, by default, can communicate to one another without explicitly specifying the domain in the address. When no messaging access restrictions are placed on addressing a Dot according to its domain (by default there are restrictions), then a Dot can also send messages to another Dot belonging to a different domain by explicitly addressing that Dot by its domain and Dot specifier.

[00285] Dots are further identified with a kind attribute as a part of the Dot Definition, and by an ID attribute. The ID is volatile and is not typically hard-coded into DHTML scripts. The addressing scheme allows for a Dot to be specified by explicitly referencing or querying a Dot's kind or ID identifier attribute.

[00286] Similarly, controls are identified by kind and by ID (also a part of the Dot Definition). Controls can also be specified by explicitly referencing or querying its kind or ID.

[00287] In the preferred embodiment, there are several permitted forms for the address of a message recipient. The fundamental one is:

#<dot specifier>:<control specifier>

Other accepted address forms are:

#<domain>:<dot specifier>
<control specifier> #<domain>:<dot kind>:<dot id>:<control kind>:<control id>
#system

If an address does not match any of these forms, the address defaults to #system.

[00288] Specifiers have four different forms: The first form specifies the unique ID of the Dot / control / domain in question. An example of this first form is:

<specifier> := <ID>

[00289] The second form allows the sender to address a message to the closest matching recipient by some form of search criteria. Multiple Dots or controls might be of the same kind, enabling broadcasting a message to these elements. An example of this second form is:

<specifier> := <kind><#search criteria>

Search criteria can be one of:

<search criteria> := any
<search criteria> := open
<search criteria> := closed

Controls that are hidden are considered closed, and controls that are visible are considered open.

[00290] The third form enables reference to a specific Dot or control, allowing messages to be sent within a Dot. If a particular control is specified, then it must be specifically referenced in the Dot. An example of this third form is:

<specifier> := dot

[00291] The final form is specifically for the control specifier. If a message is to be directed to a Dot and not a control, the control specifier must be empty. An example of this last form is:

<specifier> :=

[00292] The current embodiment supports messages to a specific recipient: the addressing/routing scheme could support messages addressed to multiple simultaneous recipients (broadcast). Similarly, Dots can address Dots on the same platform or client (computer or device); the addressing or routing scheme may also support messages addressed to Dots/Controls/Home Dot applications 200 on other devices or by user. The Home Dot application 200 routes messages sent between Dots (711, 751, 761) and resolves addressing queries (e.g., address: "#A#Any:" = first Dot of kind "A" found in the Home Dot application 200).

[00293] Since the Home Dot application 200 routes messages between Dots (711, 751, 761), it can allow or restrict Dots from addressing and sending messages to other Dots outside their own domain (the Home Dot application 200 could also restrict messages to within the same Dot). For example, Dot2 (751) is a member of domain B (750) as is Dot 3 (761), whereas Dot 1 (711) is a member of Domain A. Home Dot 200 can permit or deny messaging between Dot 1 (711) and Dots 2 and 3 (751, 761) and vice versa.

~~**[00294]** The Home Dot application 200 acts to enforce the access rights to and between published messages built on the DMA API. In one embodiment, the Home Dot application 200 accesses a database of published message methods implemented by Dot Developers and restricts/allows messages to pass based on access criteria data posted to the data base.~~

~~**[00295]** For example, a Dot developer could specify 2 public functions, 2 functions with access restricted by partner (Domain 710 or 750 for example), and 2 functions with access restricted by Dot address. The Home Dot application 200 may allow or restrict messages to be sent to one Dot from other Dots according to access criteria specified by the Dot developer.~~

~~**[00296]** The Home Dot application 200 also responds to messages sent to #system. Access to system features/functions/behaviors is one aspect of the present invention that enables extensibility of the Home Dot application system.~~

~~**[00297]** System messages include:~~

~~#REFRESH - Refreshes the user's ALL-CONFIG 311.~~

~~#INSTALL-DOT<DOT-URL> - Adds the specified Dot to the user's Home Dot application 200 (ALL-CONFIG) using the same mechanism that Dot definitions are added to the system.~~

~~#HAVE-DOT<DOT-ADDRESS> - Checks if the user currently has the specified Dot in their Home Dot application 200 (as part of the ALL-CONFIG).~~

~~#DELETE-DOT<DOT-ADDRESS> - Removes the specified Dot from the Home Dot application 200 (and the user's ALL-CONFIG).~~

~~#QUIT - Quits the Home Dot application 200.~~

~~#SAVE-TO-SERVER - Saves the user's ALL-CONFIG to the Dot application server 153.~~

~~#GET-SCREEN-WIDTH - Returns the width of the screen in pixels.~~
~~#GET-SCREEN-HEIGHT - Returns the height of the screen in pixels.~~

~~#CLOSE-ALL-DOTS - Closes all open Dots.~~

~~#OPEN-PRESET<PRESET-NAME> - Opens the specified user's Snapshot.~~
~~#TAKE-PRESET <PRESET-NAME> - Capture a Snapshot.~~

~~#GETDOT-IDS <DOT-ADDRESS> - Returns the DotID of the specified Dot.~~

~~**STATISTICS COLLECTION/ANALYSIS**~~

~~**[00298]** The Home Dot application/Dot system and method of collecting use statistics from Home Dot applications (and application servers) enables the present invention to build and query a multidimensional use-profiling database. Because Dots are used in groups and used more often for longer periods of time than web pages/sites, the present invention may collect real-time multidimensional use statistics 206 (according to which Dots are ON, for example together or simultaneously) that over time becomes a valuable multi-dimensional user behavior profiling database. The Dot server 153 collects use statistics 206 from its Home Dot application 200 on clients 199 and manages and tracks Home Dot application downloads, access to Dot Index 204, etc.~~

~~**[00299]** Information on a Dots user can be monitored in a multi-dimensional fashion. Instead of classifying a user based solely on their demographic characteristics and linear use of the internet, the present invention enables tracking on an additional dimension. A user can be classified according to which Dots they use at the same time, instead of only how they navigate within an individual viewer (the browser). This user information can be packaged and sold to content providers so that they can better provide services to their users.~~

[00300] ~~Because the present invention enables multi-dimensional profiling, a service may be provided by the operator of this system and its partners to customize advertisements and offerings to users in a more efficient and targeted manner. Tracking this profiling data in real-time allows the operator to tailor these offerings to users of specific Dots in a way that is not otherwise possible today. For example, one content provider partner can be informed that users of a certain Dot are also disproportionately monitoring content about a specific topic through another Dot. This information is then used to target ads at the moment the peripheral interests of that content providers Dot users are known. The present invention enables this information to be tracked as a complete system. The system can be utilized to facilitate serving advertisements and offerings based on the information that it collects.~~

[00301] ~~The Home Dot application 200 records application events in an XML log that intermittently gets uploaded to the Dot application server 153. Application events that are logged include the following (by timestamp and by session):~~

- ~~When a Dot is opened and when it is closed (particularly, when Home Dot 200 is opened and closed)~~
- ~~When a Dot is added from a Dot Template or a Share~~
- ~~The number of times a user clicks through from a Dot to a full-screen web page (and to which URLs)~~

- ~~The number of page views in a Dot-by-Dot session~~

- ~~When and from which URL did a user download a Home Dot application 200~~

[00302] ~~The following outlines the types of questions the use statistics of the present invention can answer regarding Home Dot application 200 and Dot use:~~

- ~~Duration and frequency of opening Home Dot applications and Dots - How often is the Home Dot application 200/Dot used on average (optionally, by user group) during the parts of a day (e.g., morning)?~~

- ~~Duration: What was the average amount of time that users leave the Home Dot application 200/Dot ON 324 during the day~~

- ~~Distribution of Home Dot application 200 and Dots (Point of Distribution including Shares) - How many users have downloaded the Home Dot application during a set period of time? From which URL were they referenced?~~

- ~~How many times has this Dot's DotTemplate been installed (from the DotIndex)?~~

- ~~From which web site were users referred? How many times has the Dot been installed from a Share?~~

- Page views in Dots, Click-throughs/URL's – How does the page-view performance of a Dot compare to the comparable content on a full-screen web site?

- List the full-screen URLs most often clicked through to (in ranking order) from this Dot.

- Dots used in Groups – Which Dots are used most frequently together? Which Dots do users most often use with Dot.

[00303] The Home Dot application 200 intermittently sends its event logs to the Dot application server 153 using the setAllEvents servlet method. The Home Dot application's events are logged and sent as an XML file in a format specified by the CALL-EVENTS.

[00304] Categorizing Dots and/or Dotcontent by "context keywords" (e.g., CNN Dot and FoxNews Dot = "News"; CBS Sportsline Dot = "Sports") enables the Home Dot application 200 to build a real-time (accessible via the DMA) multi-dimensional use context on-the-fly, based on which Dots the user currently has ON (e.g., assemble the keywords of the currently open Dots into a multi-dimensional keyword string). This highly resolved view of a user's behavior enables the Home Dot application 200 to source a very targeted offering (commerce opportunity or ad) and/or enable Dot developers to do the same. In one embodiment, the Home Dot application 200 supports

a DMA message (pay to access) that provides Dot developers with access to this type of information (e.g., SendMessage "#system" " #get-use-context-string")

[00305] The Home Dot application 200 supports a developer toolkit service that sources ads targeted according to this multi-dimensional use context generated by the Home Dot application 200 (more targeted than currently possible with singular contexts such as text strings, e.g., searched on "toys" = serve a toy ad). The Home Dot application 200 and/or Dot application server 153 monitors /tracks, and handles incremental billing for all parties (e.g., anonymously).

VARIATIONS

[00306] Many variations on the above description are contemplated and within the scope of the present disclosure. For example, in an alternate embodiment, the Home Dot application 200 interface is implemented entirely as Dot content served from Dot application server 153. Furthermore, the Home Dot application 200 itself may be packaged as a Dot. The Home Dot application Dot has special responsibilities and rights, but otherwise, it could be as much a Dot as any other Dot. This embodiment enables the application executable to be significantly smaller and the Home Dot application 200 UI to be significantly more flexible.

[00307] In yet another alternate embodiment, the Home Dot application 200 is implemented as a plug-in to the web browser.

~~**[00308]** In still another alternate embodiment, the Home Dot application 200 is implemented in Java, thus enabling it to be more easily portable to other platforms.~~

~~**[00309]** Additionally, in another embodiment, a subset of the Home Dot application 200 functionality is implemented through the browser using pop-ups; this would be particularly valuable to mobile Home Dot application 200 users who don't always have easy access to a computer with the Home Dot application 200 installed.~~

~~**[00310]** The present invention enables the definition of an interface that groups Dots together from different content providers and offers them to end users as part of a unique system of content. Content from different internet companies can be made to interact with each other using aspects of the present invention provided for in the Dot messaging architecture (DMA). The Dot server 153 may therefore be provided with functionality to broker these relationships and facilitate this interaction in Internet content between companies and between Dot-enabled internet sites.~~

~~**[00311]** Because Dots can work together (via DMA and packs), the present invention enables companies or content providers to enable their Dots to work together. The present invention provides for another layer built upon the DMA API (application programming interface) that specifies how different kinds and types of Dots from different companies or developers communicate/work together. This layer is optionally open source so that a large portion of it is self-published by the Dot Developers themselves.~~

[00312] ~~Dot Developers implement, specify, and publish in a database DMA message methods that other Dot Developers implement through calls made within their published Dots. These Dot developer methods may be aggregated and published for use in a database provided by the Home Dot publisher for example and sourced within a developer zone, or as part of a Dot Definition.~~

[00313] ~~Dot developers may implement these DMA message methods as Java script functions that other Dot Developers calls with the DMA's ##<any javascript> message. Dot Developers would need to be able to address a particular Dot's Web Conduit control that supports a page implementing the specified javascript function and address by domain if different from their domain. Restricted or variable permissions may be provided via a web server or defined in a Dot's Dot Definition to restrict/enable routing of messages.~~

[00314] ~~A Dot Developer has the option to restrict access to certain methods according to some criteria e.g., public, partners, domain. Different types of restriction criteria schemes may be implemented. A Dot may send a message to another Dot and query for methods accessible to it. An example of access levels for functions is provided below.~~

- ~~Public functions - Dot Developer implements and publishes DMA message methods that any other Dot Developer could call (e.g., what-time-is-it for a Clock Dot).~~

- ~~Reserved functions – Dot Developer implements, specifies, and publishes a set of DMA message methods that may be called only by certain types of Dot Developers (e.g. premier partner), specific Dot Developers, or specific Dots.~~

- ~~Private functions within the same domain – A Dot Developer restricts access to certain DMA message methods that may only be implemented by other Dots in the same domain. The preferred embodiment provides for the option to either allow all messages to flow between domains, or to fully restrict messages to within a domain.~~

- ~~Local Private functions within Dot – At the most restricted level, access is specified such that DMA messages may only be accessed from within the same Dot.~~

[00315] ~~A published API that specifies ways for Dots to extend functionality to other Dots enables Dot Developers to leverage the development work and functionality of other Dots. As an example, a Dot Developer may implement a credit card processing Dot. In one embodiment, this Dot is implemented as a javascript function that checks the credit available on a credit card. This developer could specify this as a public function (in the Dot Definition for example) and publish this function in a Server Dot Index. Other Dot Developers send a message to this Dot and call "check-credit", thus leveraging the development carried out from the first Dot Developer.~~

~~**[00316]** In the above example, the messaging architecture may provide revenue generation by way of monthly fees, per-access fees, etc. The Home Dot application 200 and/or Dot application servers could monitor/track and automatically handle incremental billing for all parties. By applying this method to the previous example, a Dot Developer who calls a credit card processing Dot for a credit check may be incrementally billed for each such call.~~

~~**[00317]** It will be appreciated that the methods, in the form of instructions having a sequence, syntax, and content, of the present invention may be stored on (or equivalently, in) any of a wide variety of computer-readable media such as magnetic media, optical media, magneto-optical media, electronic media (e.g., solid state ROM or RAM), etc., the form of which media not limiting the scope of the present invention. A data processor reading said media is operable to either transfer (e.g., download) said instructions thereto and then operate on those instructions, or cause said instructions to be read from the media and operate in response thereto. Furthermore, devices (e.g., a reader) for accessing the instructions on said media may be contained within or connected directly to the data processor residing on a device on which those instructions operate, or may be connected via a network or other communication pathway to said data processor.~~

~~**[00318]** While a plurality of preferred exemplary embodiments have been presented in the foregoing detailed description, it should be understood that a vast number of variations exist, and these preferred exemplary embodiments are merely representative examples, and are not intended to limit the scope, applicability or~~

~~configuration of the invention in any way. Rather, the foregoing detailed description provides those of ordinary skill in the art with a convenient guide for implementation of the invention, by way of examples, and contemplates that various changes in the functions and arrangements of the described embodiments may be made without departing from the spirit and scope of the invention defined by the claims thereto.~~

~~Appendix A~~

~~This appendix contains an exemplary definition of an XML structure used for Dots. Generally, there are four top XML structures that are read from URL/file. These are:~~

~~Session~~	~~<SESSION_CONFIG>~~
~~User configuration~~	~~<ALL_CONFIG>~~
~~shares~~	~~<SHARE>~~
~~Dot templates~~	~~<DOT>~~

~~These structures are described below. Some structures reuse other structure formats.~~

<DOT>

~~GLOBALID= "String"~~

~~Used only within <SHARE> tags. This GLOBALID must be unique with respect to other <DOT> tags in the <SHARE>.~~

~~DOMAIN = "string"~~

~~Unique label for the owning company of this Dot. In one embodiment, Dots are limited to communicating with Dots only from their own domain.~~

~~KIND= "String"~~

~~Helpful identifier for finding such a Dot from another Dot. Does not have to be unique.~~

~~<FRAME>~~

~~<TITLE>~~

~~TEXT = "string"~~

~~JUSTIFY= "LEFT" | "CENTER" | "RIGHT"~~

~~COLOR= "#XXXXXX"~~

~~PIXELWIDTH="integer"~~

~~Width in pixel units. Overrides WIDTH attribute.~~

~~**PIXELHEIGHT="integer"**~~

~~Height in pixel units. Overrides HEIGHT attribute.~~

~~**WIDTH="Integer"**~~

~~Width in Dot units. Default value is 1.~~

~~**HEIGHT="integer"**~~

~~Height in Dot units. Default value is 1.~~

~~**X="integer" | "LEFT" | "CENTER" | "RIGHT"**~~

~~Initial X position in screen coordinates. Default is center.~~

~~**Y= "INTEGER" | "TOP" | "CENTER" | "BOTTOM"|**~~

~~Initial Y position in screen coordinates. Default is center.~~

~~**FIXWIDTH="TRUE" | "FALSE"**~~

~~Default is false.~~

~~**FIXWIDTH="TRUE" | "FALSE"**~~

~~Default is false.~~

~~**BACKGROUNDCOLOR="#XXXXXX"**~~

~~Default is white.~~

~~**<TITLEBARIMAGE>**~~

~~JUSTIFY =“LEFT” | “CENTER” | “RIGHT”~~

~~TILELEFT=“integer”~~

~~TILERIGHT= “integer”~~

~~NORMAL= “url”~~

~~DOWN= “URL”~~

~~HOVER=“URL”~~

~~INACTIVE=“URL”~~

~~**<BOTTOMARIMAGE>**~~

~~JUSTIFY= “LEFT” | “CENTER” | “RIGHT”~~

~~TILELEFT= “integer”~~

~~TILERGHT =“integer”~~

~~NORMAL= “URL”~~

~~DOWN=“URL”~~

~~HOVER=“URL”~~

~~INACTIVE=“URL”~~

~~**<MENU>**~~

~~Contains zero or more <ITEM> tags.~~

~~**<ITEM>**~~

~~TITLE= “string”~~

~~TOOLTIP=“string”~~

~~ICON=“URL”~~

~~ID=“string”~~

~~Must be unique.~~

~~**<ACTION>**~~

~~RECIPIENT="address"~~

~~MESSAGE="string"~~

~~**<CONTROL_LAYOUT>**~~

~~Contains zero or more <CONTROL> tags.~~

~~**WIDTH=** "integer"~~

~~Divides control space into this many evenly-spaced columns. Default is 1.~~

~~**HEIGHT="integer"**~~

~~Divides control space into this many evenly-spaced rows. Default is 1.~~

~~**WIDTHSCALES="TRUE" | "FALSE"**~~

~~Default is true.~~

~~**HEIGHTSCALES="TRUE" | "FALSE"**~~

~~Default is true.~~

~~**<CONTROL>**~~

~~See external definition.~~

~~**<CATEGORIES>**~~

~~Contains zero or more <CATEGORY> tags.~~

~~**<CATEGORY>**~~

~~Adds the Dot to this category. This is the only way categories are specified; i.e. there is no master category list.~~

~~**NAME="string"**~~

~~This is the name of the category.~~

~~**<EVENTS>**~~

~~**<ONCLOSE>**~~

~~Executes this action list on close.~~

~~<ACTION LIST>~~

~~See external definition.~~

<CONTROL>

~~The only understood value for this is "Browser". Currently, any other value is overridden and "Browser" is assumed. All specifications should explicitly specify "Browser".~~

~~**KIND="string"**~~

~~**ID="string"**~~

~~Must be unique with other controls in this Dot.~~

~~**LEFT="integer"**~~

~~X position in container units. Default is **0**.~~

~~**TOP="integer"**~~

~~Y position in container units. Default is **0**.~~

~~**WIDTH="integer"**~~

~~Width in container units. Default is 1~~

~~**HEIGHT=" integer"**~~

~~Height in container units. Default is 1.~~

~~**URL="URL"**~~

~~This is read **if** and only **if** this control is of class "Browser". This is the URL to which this control navigates.~~

<PRESET>

~~Contains zero or more <DOT> tags.~~

~~**TITLE="string"**~~

~~<DOT>~~

~~This tag specification is specific to the <PRESET> tag, i.e. it is not the same tag as the full <DOT> specification described elsewhere.~~

~~**GLOBALID="string"**~~

~~Used only within <SHARE> tags. This <SHARE> must contain a <DOT> with a matching GLOBALID.~~

~~**X="integer" | "LEFT" | "CENTER" | "RIGHT"**~~

~~**Y= "integer" | "TOP" | "CENTER" | "BOTTOM"**~~

<SHARE>

~~Contains zero or more <DOT> and <PRESET> tags. GLOBALID attributes are valid and consistent only within this share.~~

~~**<DOT>**~~

~~See external definition.~~

~~**<PRESET>**~~

~~See external definition.~~

<ACTION LIST>

~~Contains zero or more <ACTION> tags.~~

~~**<ACTION>**~~

~~**RECIPIENT="address"**~~

~~**MESSAGE= "string"**~~

<SESSION CONFIG>

~~**REDIRECT="URL"**~~

~~The client reading this will ignore this configuration and try again with the configuration stored at this URL. The client will not redirect if this tag is omitted, if the URL is empty, or if the URL is the same URL used to retrieve this configuration in the first place. This step is recursive, so if two configuration files are redirecting to each other, the client will get stuck in an endless loop.~~

~~**VERSION="string"**~~

~~This attribute is required and must match the version of the client. Currently, the client is compiled with the version "alpha:1". If this attribute is omitted or differs from the client's version, the client will download the executable specified in the UPGRADE attribute and execute it.~~

~~**UPGRADE="URL"**~~

~~This is used only if the VERSION attribute differs from the client's version. This URL specifies the location of an executable that will upgrade the client's application. The client will quit, and then run the upgrade executable automatically.~~

~~**<Methods>**~~

~~**Basecurl="url"**~~

~~All the tags encapsulated here represent servlet methods. The URLs specified by the method are relative to this base URL.~~

~~**<ADD_USER>**~~

~~URL="URL"~~

~~**<GET_USER>**~~

~~URL="URL"~~

~~**<SET_USER>**~~

~~URL="URL"~~

~~**<GET_SESSION_CONFIG>**~~

~~URL="URL"~~

~~<SET_PASSWORD >~~

~~URL="URL"~~

~~<GET_MASTER_DOT_LIST>~~

~~URL="URL"~~

~~<SET_ALLCONFIG>~~

~~URL="URL"~~

~~<GET_ALL_CONFIG>~~

~~URL="URL"~~

~~<SET_ALL_STATS>~~

~~URL="URL"~~

~~<GET_DOT_TEMPLATE>~~

~~URL="URL"~~

~~<ADD_SHARE>~~

~~URL="URL"~~

~~<GET_SHARE>~~

~~URL="URL"~~

~~<AUTH_TEST>~~

~~URL="URL"~~

~~<DO_REDIR>~~

~~URL="URL"~~

~~<SKELETONS>~~

~~<HOMEDOT>~~

~~<IMAGES>~~

~~BASEURL="URL"~~

~~This is the **URL** of the root directory for where the HomeDots images are stored.~~

~~<DOT>~~

~~<IMAGES>~~

~~BASEURL="URL"~~

~~This is the URL of the root directory for where the default images for Dots are stored.~~

~~<DOTS>~~

~~<ADD>~~

~~TEMPLATE="URL"~~

~~This is the URL for the XML specification of the "add" Dot.~~

~~<HELP>~~

~~TEMPLATE="URL"~~

~~This is the URL for the XML specification of the "help" Dot.~~

~~<LOGIN>~~

~~TEMPLATE="URL"~~

~~This is the URL for the XML specification of the "login" Dot~~

<ALL_CONFIG

~~<SHARE>~~

~~This contains all the installed Dots and presets in this users configuration. See external definition.~~

~~**<LASTSTATE>**~~

~~**<PRESET>**~~

~~This preset uses the GLOBALID values in the above <SHARE>. This represents the last known position of the open Dots when the user last quit the application.~~

~~**<HOMEDOT>**~~

~~This represents the last position of the HomeDot when the user quit his application.~~

~~X="integer" | "left" | "center" | "right"~~

~~Y="integer" | "top" | "center" | "bottom"~~

~~**Appendix B**~~

~~Client-Server Protocols~~

~~HTTP/HTTPS~~

~~HTTP/HTTPS is the chosen basic request response protocol for client/server communications for Dots. HTTP/HTTP by default runs on port 80/443 and is typically left unblocked on the firewall by most network administrators (and by default by the firewall vendors). It is also the logical choice given that web servers and web client libraries are cheap and often freely available. This choice effectively removes the burden of creating and maintaining code to the low level TCP/IP socket connections as well as the basic request response framework. For security reasons, it is suggested that all user-specific data exchanges be done through HTTPS (as opposed to HTTP).~~

~~XML~~

~~XML is the chosen data exchange format between the client and the server.~~

Session_Config DTD

```
<!ELEMENT session_config
        add_user,
        set_password, set_session_config,
        get_master_dot_list,
        get_dot_template,
        get_all_config,
        get_all_confg,
        set_all_events,
        dotdots_navigation
>

<!ELEMENT add_user EMPTY>
<!ATTLIST add_user
        url     CDATA       #REQUIRED
>

<!ELEMENT set_password EMPTY>
<!ATTLIST set_password
        url     CDATA       #REQUIRED
>

<!ELEMENT get_session_config EMPTY>
<!ATTLIST get_session_config
        url     CDATA       #REQUIRED >
>

<!ELEMENT get_master_dot list EMPTY>
<!ATTLIST get_master_dot_list
        url     CDATA       #REQUIRED
        refresh CDATA       #REQUIRED
>

<!ELEMENT get_dot_template EMPTY>
<!ATTLIST get_dot_template
        url     CDATA       #REQUIRED
>
```

~~<!ELEMENT get_all_config EMPTY>~~
~~<!ATTLIST get_all_config~~
~~url CDATA #REQUIRED~~
~~>~~

~~<!ELEMENT set_all_config EMPTY>~~

<!ATTLIST set_all_config
url CDATA #REQUIRED
refresh CDATA #REQUIRED
>

<!ELEMENT set_all_events EMPTY>
<!ATTLIST set_all_events
url CDATA #REQUIRED
>

<!ELEMENT dodots_navigation EMPTY>
<!ATTLIST dodots_navigation
url CDATA #REQUIRED
>

Session_Config Example

<?xml version="1.0"?>
<!DOCTYPE SESSION_CONFIG SYSTEM "SESSION_CONFIG.DTD">
<SESSION_CONFIG>
<ADD_USER
url= http colon slash slash scot colon 8080 slash servlet slash DoDotsServer slash addUser quote space slash greater than
<SET_PASSWORD url=http colon slash slash scot colon 8080 slash servlet slash DoDotsServer slash setPassword quote space slash greater than
<GET_SESSION_CONFIG
url= http colon slash slash scot colon 8080 slash servlet slash DoDotsServer slash getSessionConfig quote space slash greater than
refresh="1" />
<GET MASTER DOT LIST url= /> http colon slash slash scot colon 8080 slash servlet slash DoDotsServer slash getMasterDotList quote space slash greater than
refresh=" 1" />
<GET_DOT_TEMPLATE url= http colon slash slash scot colon 8080 slash servlet slash DoDotsServer slash getDotTemplate quote space slash greater than
refresh="1" />
<GET_ALL_CONFIG url= http colon slash slash scot colon 8080 slash servlet slash DoDotsServer slash getAllConfig quote space slash greater than
refresh="600"/>
<SET_ALL_EVENTS
url= http colon slash slash scot colon 8080 slash servlet slash DoDotsServer slash setAllEvents quote
refresh="600" />

<DODOTS_NAVIGATION url=
http colon slash slash www dot DoDots dot com slash quote greater than
</SESSION CONFIG>

User DTD

<!ELEMENT user EMPTY>
<!ATTLIST user

uid	CDATA	#IMPLIED
password	CDATA	#IMPLIED
dpid	CDATA	#IMPLIED
title	CDATA	#IMPLIED
prefix	CDATA	#IMPLIED
firstname	CDATA	#IMPLIED
mi	CDATA	#IMPLIED
lastname	CDATA	#IMPLIED
suffix	CDATA	#IMPLIED
email	CDATA	#IMPLIED
street1	CDATA	#IMPLIED
street2	CDATA	#IMPLIED
city	CDATA	#IMPLIED
state	CDATA	#IMPLIED
country	CDATA	#IMPLIED
birthdate	CDATA	#IMPLIED
sex	CDATA	#IMPLIED

>

User Example

<?xml version="1.0" ?>
<GETDOCTYPE USER SYSTEM "USER.DTD">
<USER>
uid ="joeblack"
password ="morte"
dpid ="321"
title ="Chief Dot Developer"
prefix ="Mr." firstname ="Joe"
mi ="D"
lastname ="Black"
suffix ="Sr."
email ="reaper@otherside.org"
street1 ="1 KickTheBucket Street"
street2 =""
city ="Mordor"

state="DC"
country=""
birthdate=""
sex="N"
/>

Dot_Master DTD

<!ELEMENT dot_master(
dot+
)
>
<!ATTLIST dot_master
>
<!ELEMENT dot EMPTY>
<!ATTLIST dot
dtid ID #REQUIRED
name CDATA #REQUIRED
desc CDATA #IMPLIED
>

Dot_Master Example

<?xml version="1.0" ?>
<GETDOCTYPE DOT_MASTER SYSTEM "DOT_MASTER.DTD">
<DOT_MASTER>
<DOT DTID="1" NAME="Scot's DoDots stock tracker" DESC="Tracks and graphs the stock performance of the DoDots corporation"
/>
<DOT DTID="32" NAME="ACME Wonder Dot" DESC="The most amazing dot in the world. This one does tricks like you've never seen before"
/>
<DOT DTID="33" NAME="ACME Wonder Dot version 2" DESC="The most amazing Dot in the world. This one actually works the way the first one was supposed to."
/>
<DOT DTID="40" NAME="Roadrunner's Coyote trap watcher" DESC="Up to the minute alerts on new traps set by all coyotes within a 100 mile radius"
/>
</DOT MASTER>

Dot_Template DTD

```
<!ELEMENT dot_template (
menu*,
        controls
)
>
<!ATTLIST dot
    uid                 ID                              #REQUIRED
    debug               ("TRUE"|"FALSE")        "FALSE"
    type                ("ACTIVEX"|"JAVA")      "ACTIVEX"
    class               CDATA                           #REQUIRED
    title               CDATA                           #REQUIRED
    titlebarimage       CDATA                           #REQUIRED
    bottombarimage      CDATA                           #REQUIRED
    width               CDATA                           #REQUIRED
    height              CDATA                           #REQUIRED
    resizable           ("TRUE"|"FALSE")        "FALSE"
>
<!ELEMENT menu
        ( menuitem+
        )
>
<!ATTLIST menu
        id      ID #REQUIRED
>

<!ELEMENT controls (
        control+
        )
>
<!ATTLIST controls
layouts             ("flow"         #IMPLIED "flow"
                    |"grid"
                    |"gridbag")
>

<!ELEMENT menuitem EMPTY>
<!ATTLIST menuitem
        id          ID                      #REQUIRED
        title       CDATA                   #REQUIRED
        icon        CDATA                   #IMPLIED
        tooltip     CDATA                   #IMPLIED
        enabled     ("TRUE"|"FALSE")    #IMPLIED "FALSE"
        action      CDATA                   #REQUIRED
>
```

```
<!ELEMENT control EMPTY>
<!ATTLIST control
        id          ID                          #REQUIRED
        type        ("ACTIVEX"| "JAVA")  "ACTIVEX"
        class       CDATA                       #REQUIRED
        width       CDATA                       #REQUIRED
        height      CDATA                       #REQUIRED"
        location    CDATA                       #REQUIRED
>
```

Dot_Template Example

```
<?xml version="1.0"?
<!GETDOCTYPE DOT_TEMPLATE SYSTEM "DOT_TEMPLATE.DTD">
<DOT_TEMPLATE
        DEBUG="TRUE"
        TYPE="ACTIVEX"
        CLASS=" 12939-292939-93939-29292"
        TITLE="Ebay"
        TITLEBARIMAGE= http colon slash slash www dot ebay dot com slash image slash
        foo quote
        BOTTOMBARIMAGE="someurl"
        WIDTH=2
        HEIGHT=1
        RESIZABLE="TRUE"
>
        <MENU>
                <MENUITEM
                        ID="NEW_AUCTION_WINDOW"
                        TITLE="New Auction Window"
                        ICON=" quote http colon slash slash www dot ebay dot com
                        slash images slash new underscore auction dot gif quote
                        TOOLTIP="Open up a new auction window to use"
                        ACTION="someaction.exe"
                />
                <MENUITEM
                        ID="DISPLAY_CLOSE"
                        TITLE="Close display"
                        ICON="http colon slash slash www dot ebay dot com slash images
                        slash menuitem dot gif quote
                        TOOLTIP="Close the display field"
                        ACTION="someaction.exe"
                />
        </MENU>
```

~~<CONTROLS LAYOUT="FLOW">~~
~~<CONTROL~~
~~ID="SEARCH FIELD"~~
~~TYPE="ACTIVEX"~~
~~CLASS="33333-1111111-222222-33333"~~
~~WIDTH="80"~~
~~HEIGHT="20"~~
~~LOCATION="TOP"~~
~~URL=http colon slash slash www dot webster dot com slash searchField dot html~~
~~/>~~
~~<CONTROL~~
~~ID="RESULT_DISPLAY"~~
~~TYPE="ACTIVEX"~~
~~CLASS="33333-1111111-222222-33333" WIDTH="80"~~
~~HEIGHT="100"~~
~~LOCATION="TOP"~~
~~VISIBILITY="FALSE"~~
~~/>~~
~~</CONTROLS>~~
~~</DOT TEMPLATE>~~

All_Config DTD

```
<! DTD for the All_Config element used in the configuration of DoDots's Snapshot,
block and dot objects  >

<!ELEMENTS all_config (
        Snapshot*
        )
>
<!ATTLIST all_config
>
<!ELEMENT Snapshot (
        block*,
        dot*
        )
>
<!ATTLIST Snapshot
        id              ID      #REQUIRED
>

<!ELEMENT block(
        dot*
        )
>

<!ELEMENT dot (
        menu*,
        controls
        )
>
<!ATTLIST dot
        id              ID                      #REQUIRED
        debug           ("TRUE"|"FALSE")        "FALSE"
        type            ("ACTIVEX"|"JAVA")      "ACTIVEX"
        class           CDATA                   #REQUIRED
        title           CDATA                   #REQUIRED
        titlebarimage   CDATA                   #REQUIRED
        bottombarimage  CDATA                   #REQUIRED
        width           CDATA                   #REQUIRED
        height          CDATA                   #REQUIRED
        resizable       ("TRUE"|"FALSE")        "FALSE"
```

```
<!ELEMENT menu (
        menuitem+
        )
>
<!ATTLIST menu
        id      ID      #REQUIRED
>
<!ELEMENT controls (
        control+
        )
>
<!ATTLIST control
layout          (“flow”         #IMPLIED “flow”
                |“grid”
                |“gridbag”
>

<!ELEMENT menuitem EMPTY>
<!ATTLIST menuitem
        id              ID                              #REQUIRED
        type            (“ACTIVEX”|“JAVA”)      “ACTIVEX”
        class           CDATA                           #REQUIRED
        width           CDATA                           #REQUIRED
        height          CDATA                           #REQUIRED
        location        CDATA                           #REQUIRED
>
```

All_Config Example

```
<?xml version=“1.0”?>
<!DOCTYPE ALL_CONFIG SYSTEM “ALL_CONFIG.DTD”>
<ALL_CONFIG>
        <SNAPSHOT ID=“1”
                        <DOT
                                DEBUG=“TRUE”
                                TYPE=“ACTIVEX”
                                CLASS=“ 12939-292939-93939-29292”
                                TITLE=“Ebay”
                                TITLEBARIMAGE= http colon slash slash www dot ebay dot
                                com slash image slash foo quote
                                BOTTOMBARI MAGE=“someurl”
                                WIDTH=2
                                HEIGHT=1
                                RESIZABLE=“TRUE”
                >
```

~~<MENU>~~
~~<MENUITEM~~
~~ID="NEW_AUCTION _ WINDOW"~~
~~TITLE="New Auction Window"~~
~~ICON=" http colon slash slash www dot ebay dot com slash images slash new underscore auction dot gif quote~~
~~TOOLTIP="Open up a new auction window to use"~~
~~ACTION="someaction.exe"~~
~~/>~~
~~<MENUITEM~~
~~ID="DISPLAY_CLOSE"~~
~~TITLE="Close display"~~
~~ICON=" http colon slash slash www dot ebay dot com slash images slash menuitem dot gif quote~~
~~TOOLTIP="Close the display field"~~
~~ACTION="someaction.exe~~
~~/>~~
~~</MENU>~~
~~<CONTROLS LAYOUT="FLOW">~~
~~<CONTROL~~
~~ID="SEARCH_FIELD"~~
~~TYPE="ACTIVEX"~~
~~CLASS="33333-1111111-222222-33333"~~
~~WIDTH="80"~~
~~HEIGHT="20"~~
~~LOCATION="TOP"~~
~~URL=http colon slash slash www dot webster dot com slash searchField dot html~~
~~/>~~
~~<CONTROL~~
~~ID="RESULT_DISPLAY"~~
~~TYPE="ACTIVEX"~~
~~CLASS="33333-1111111-222222-33333"~~
~~WIDTH="80"~~
~~HEIGHT="100"~~
~~LOCATION="TOP"~~
~~VISIBILITY="FALSE"~~
~~/>~~
~~</CONTROLS>~~
~~</DOT >~~
~~</SNAPSHOT>~~
~~</ALL CONFIG>~~

All_Events DTD

```
<!ELEMENT dot_events
        dot_event*
>
<!ATTLIST dot_events
>

<!ELEMENT dot_events EMPTY>
<!ATTLIST dot_events
        deid            CDATA           #REQUIRED
        dtid            CDATA           #IMPLIED
        dpkid           CDATA           #IMPLIED
        dpid            CDATA           #IMPLIED
        start_time      CDATA           #REQUIRED
        end_time        CDATA           #IMPLIED
        url1            CDATA           #IMPLIED
>
```

NOTES:

All data/time data transmitted must conform to the W3C HTTP1.1 specification's HTTPdate grammar (IETF RFC 1123 format is strongly preferred). This means, among other things, that all date/time data will be represented in GMT/UTC time. See RFC 1123 for specific details. Simplified format string:

Wdy, dd MON yyyy HH:MM:SS GMT

The valid even ids (deid) will be stored in the DOT_EVENTS table described later in this appendix.

- 0 - Download of homedot/native code
- 1 - Download of Dot
- 2 - download of Dotpack
- 3 - Use of Dot (home Dot is special dtid " 1" for this event)
- 4 - Click thru to browser
- 5 - Page view

DEID and start_time are always required, but other required attributes are actually variable based on record type/deid:

- 0 -dpid, end_time
- 1 -dpid, dtid, end, time
- 2 -dpid, dpkid, end_time
- 3 -dtid, end_time
- 4 -dtid, url1
- 5 -dtid

All_Events Example

<DOT_EVENTS>
<DOT_EVENT DEID="0" DPID="1 0" START_TIME="Sat, 10 APR 1999 09:15:08 GMT" END_TIME="Sat, 10 APR 1999 09:16:55 GMT" />
<DOT_EVENT DEID="2" DPKID="5" DPID="1 0" START_TIME="Sat, 10 APR 1999 09:17:08 GMT" END_TIME="Sat, 10 APR 1999 09:19:55 GMT" />
<DOT_EVENT DEID="1" DTID="49" DPID="1" START_TIME="Sat, 10 APR 1999 09:30:08 GMT" END_TIME="Sat, 10 APR 1999 09:30:55 GMT" />
<DOT_EVENT DEID="1" DTID="51" DPID="1" START_TIME="Sat, 10 APR 1999 09:31:08 GMT" END_TIME="Sat, 10 APR 1999 09:32:55 GMT" />
<DOT_EVENT DEID="1" DTID="59" DPID="1" START_TIME="Sat, 10 APR 1999 09:33:08 GMT" END_TIME="Sat, 10 APR 1999 09:34:55 GMT" />
<DOT_EVENT DEID="3" DTID="49" START_TIME="Sat, 10 APR 1999 09:35:08 GMT" END_TIME="Sat, 10 APR 1999 12:29:55 GMT" />
<DOT_EVENT DEID="3" DTID="51" START_TIME="Sat, 10 APR 1999 09:35:08 GMT" END_TIME="Sat, 10 APR 1999 12:29:55 GMT" />
<DOT_EVENT DEID="3" DTID="59" START_TIME="Sat, 10 APR 1999 09:35:08 GMT" END_TIME="Sat, 10 APR 1999 12:29:55 GMT" />
<DOT EVENT DEID="5" DTID="59" START TIME="Sat, 10 APR 1999 09:15:08 GMT" />
<DOT EVENT DEID="5" DTID="59" START TIME="Sat, 10 APR 1999 11:34:08 GMT" />
<DOT_EVENT DEID="5" DTID="59" START_TIME="Sat, 10 APR 1999 12:15:08 GMT" />
</DOT_EVENTS>

AddShare_Response DTD

```
<!ELEMENT addshare_response EMPTY>
<!ATTLIST addshare_response
        dusid          CDATA      #REQUIRED
>
```

AddShare_Response example

```
<ADDSHARE_RESPONSE DUSID="123456" />
```

8 Server API

As indicated previously in this document, the client communicates to the server via an HTTP request. The HTTP request URL will effectively contain an "API call". Any API calls that require the uploading of data to the server will place the data in the message body of the request. This implies that these (uploader) HTTP requests will be POSTs and not GETs. API calls that require uploading/downloading of data will send/receive that data in XML, in the format as is indicated previously in this document. The server will support message body compression (i.e., content-encoding) to help reduce network traffic. Content compression will be in the commonly supported HTTP forms of :gzip, zip, or zlib and will be chosen on a user by user basis based on standard HTTP response and request header values for content encoding.

All "API Ccalls" listed below are the end part of a complete URL that starts with: http colon slash slash open bracket someservername close brackt dot DoDots dot com slash DoDots slash, where <servername> is variable and DoDots is an alias for the DoDots servlet/JSP servlet or servlets directory. Arguments to any of the methods are passed in as name value pairs in the query string portion of the URL. Note that data intended to be persistently stored on the server are not considered "arguments" to the server API method. The arguments are used only to allow the servlet implementing the API call to perform the requested action, not to encode the data uploaded.

In every case, the client will receive an HTTP response code of: "401 Not Authorized" if the client does not provide a valid username and password in every request. To avoid the possibility of a username/password combination persistently stored in the clear (i.e., a directory entry on windows), the standard HTTP headers of "Authorization:" and "WWW-Authenticate:" will be used to carry the username and password information in every request with the exception of the DoAddUser method. The security scheme used will be the "BASIC" scheme as detailed in IETF RFC 1521 and RFC 2617. This has the two added

advantages of: a) an ordinary web browser can then be used as an impromptu debugging tool for the client/server communication; and, b) user sessions can be silently reestablished when the session has timed-out due to inactivity and the user causes a call to the server to happen without restarting the client WinInet, like MSIE, will keep resending the Authorization header with every request until the hosting application exits. It is suggested that the application server handle the authentication via the HTTP header information rather than configuring the web server for authentication - this way the authentication scheme can change over time by module replacement (both on the server and the client) and the user can be offered the ability to change passwords. The BASIC authentication method is recommended for the initial release due to the readily available BASE64 decoders, including the built-in support for BASIC auth in most client-side libraries (like WinInet). Other, more secure authentication schemes (such as MD5 or X509) may be substituted or added in, if desired. All successful requests will return a standard HTTP response with a result code of "200 OK" unless otherwise specified in the method descriptions below. The server should pay careful attention to the HTTP "content-length" and the actual request body lengths to confirm that the client message has not been truncated or the connection dropped. The server may be designed and implemented with localization and internationalization of returned error, warning and status messages. Given that certain XML streams may be of great length, the server should support inbound and outbound handing of message body compression/decompression when requested by the client. Compression types that should be supported are: GZIP, ZIP, and ZLIB. The client indicates requested/supported compression types thru the standard HTTP headers of "Content-Encoding", "Accept-Encoding", and "Vary" as described in the W3C HTTP 1.1 specification. Note that the client may indicate or not indicate compression on a request by request basis. The server should not assume that the client can handle a compressed response body unless indicated in the request header.

Server API methods for the end-user client

addUser

The add User method is used to create a new DoDots user account.

Note that this is available as an end-user api in cases in which it is desired to allow new users to add themselves rather than wait for an administrator to do it for them.

Arguments:

- uid - unique username
- password - password for the user account
- dpid (optional) - Dot provider id user is also a Dot content developer
- title (optional) - title if user is also a Dot content developer
- prefix (optional) - name, prefix (i.e., Mr., Ms., Mrs., Dr., etc.)

- ~~firstname - First name~~
- ~~mi (optional) - Middle initial~~
- ~~lastname - Last name~~
- ~~suffix (optional) - name suffix (i.e. III, Jr., PhD., etc.)~~
- ~~email (optional) - E-mail address~~
- ~~street1 (optional) - first line of the street address~~
- ~~street2 (optional) - second line of street address~~
- ~~city (optional) - city~~
- ~~state (optional) - state~~
- ~~country (optional) - country~~
- ~~birthdate (optional) - birthdate (i.e. 04/10/1965)~~
- ~~sex (optional) - sex (i.e. male, female)~~

~~Example:~~
~~Request:~~
~~http colon slash slash dots1 dot DoDots dot com slash DoDots slash addUser question mark uid equal Smith ampersand password equal scotchromanian ampersand prefix equal Mr ampersand firstname equal Scot ampersand mi equal J ampersand lastname equal Smith ampersand email equal Smith ampersand newco.com~~

~~Response:~~
~~Standard HTTP success with no response body.~~

~~getUser~~
~~The getUser method retrieves the user's own demographic data. This method supports the ability of the client application to then allow updates to the user's demographic information using the setUser method (see below).~~

~~Arguments:~~

- ~~uid (optional) - username. uid is required when the issuer is a Dot developer or the DoDots admin. If the method is invoked by an end-user, this argument is ignored and the server identifies the user from the Authorization information.~~

Example:
Request:
http colon slash slash dots1 dot DoDots dot com slash DoDots slash getUser

Response:
The expected response is an xml stream containing the user's current demographic information. This xml stream is the USER DTD as defined earlier in this document.

setUser

- uid (optional) - username. uid is required when the issuer is a Dot developer or the DoDots admin. If the method is invoked by an end-user, this argument is ignored and the server identifies the user from the Authorization information.

- password - password for the user account

- dpid (optional) - Dot provider id user is also a Dot content developer

- title (optional) - title if user is also a Dot content developer

- prefix (optional) - name, prefix (i.e., Mr., Ms., Mrs., Dr., etc.)

- firstname - First name

- mi (optional) - Middle initial

- lastname - Last name

- suffix (optional) - name suffix (i.e. Ill, Jr., PhD., etc.)

- email (optional) - E-mail address

- street1 (optional) - first line of the street address

- street2 (optional) - second line of street address

- city (optional) - city

- state (optional) - state

- country (optional) - country

- birthdate (optional) - birthdate (i.e. 04/10/1965)

- sex (optional) - sex (i.e. male, female)

Example:
Request:
http colon slash slash dots1 dot DoDots dot com slash DoDots slash setUser question mark uid equal Smith ampersand password equal scotchromanian ampersand prefix equal Mr ampersand firstname equal Scot ampersand mi equal J ampersand lastname equal Smith ampersand email equal smith at newco.com

Response:
Standard HTTP success with no response body.

getSessionConfig

The getSessionConfig method is used to retrieve general configuration information.
Arguments:

- None

Example:
Request:
http colon slash slash dots1 dot DoDots dot com slash DoDots slash getSessionConfig

Response:
The expected response is an xml stream containing the high level configuration information for the session. This xml stream is the SESSION_CONFIG DTD as defined earlier in this document.

setPassword

The setPassword method provides a mechanism for the user to change their associated DoDots server account password.

Arguments:
Newpass – the new password

Example:
Request:
http colon slash slash dots1 dot DoDots dot com slash DoDots slash setPassword question mark newpass equal bar

Response if successful:
Standard HTTP success with no response body.

getMasterDotList

The getMasterDotList method requests the list of all Dot templates known by the server. The server returns a message body in predefined XML format (set the DOT/MASTERLIST DTD and example) of all possible Dots and their default template values. Note that included in

each DOT element are a version stamp and Dot class ids which allows the client to determine if they have the most up-to-date Dot template and Dot class binaries for the dot. The client may then decide to request the updated Dot template (see the getDotTemplate method) and/or Dot class binary zip file (see the getDotClassBinaries method).

Arguments:
- None.

Example:
Request:
http colon slash slash dots1 dot DoDots dot com slash DoDots slash getMasterDotList

Response:
Standard HTTP success with response body as an xml stream in the predefined XML format of the DOT/MASTERLIST DTD defined earlier in this document.

setAllConfig

The setAllConfig method sends the entire body of user-specific client-side configuration information (all Snapshots, blocks, Dots, general settings, etc.) to the server for persistent storage on behalf of the user. The data is sent as the message body in a predefined XML format (see the ALL_CONFIG DTD and example). The response returned by the server indicates success or failure.

Arguments:

- none (request body contains the xml stream in the predefined XML format of the ALL_CONFIG DTD defined earlier in this document.)

Example:
Request: http colon slash slash dots1 dot DoDots dot com slash DoDots slash set AllConfig

Response:
Standard HTTP success with no response body.

getAllConfig

The getAllConfig method retrieves the last user specific, server side saved, client side configuration information (all Snapshots, blocks, Dots, general settings, etc.) from the server. The data is sent as the message body in a predefined XML format (see the ALL_CONFIG DTD and example).

Arguments:
- **none**

Example:
Request:

http colon slash slash dots1 dot DoDots dot com slash DoDots slash getAllConfig

Response:
Standard HTTP success with response body as an xml stream in the predefined XML format of the ALL_CONFIG DTD defined earlier in this document.

setAllEvents

The setAllEvents method sends usage information that has accumulated since the last call to setAllEvents to the server for later use in statistical usage report generation. The data is sent as the message body in a predefined XML format (see the ALL_EVENTS DTD and example). The response returned by the server indicates success or failure.

Arguments:
- none

Example:
Request:
http colon slash slash dots1 dot DoDots dot com slash DoDots slash setAllEvents

Response:
Standard HTTP success with no response body.

getDotTemplate

The getDotTemplate method asks the server for a specific Dot template. The server returns the XML that describes the default state of the Dot. This method is called when the client has determined that it does not have the current version of the Dot and the user wants the most recent version (NOTE: this includes when the user is downloading the Dot for the first time)

Arguments:
- DTID – The unique Dot id for the Dot being requested. This value is extracted from the Dot master list xml stream returned by the getMasterDotList method for each Dot.

Example:
Request:
http colon slash slash dots1 dot DoDots dot com slash DoDots slash getDotTemplate question mark dtid equal 1234

Response:
Standard HTTP success with response body as an xml stream in the predefined XML format of the DOT TEMPLATE DTD defined earlier in this document.

addShare

The addShare method stores a specific shared Dot, pack, or Snapshot. The server takes the XML that was uploaded by the client for sharing and returns the unique id of the share The client should put the XML stream to be shared in the request body.

Arguments:
- none

Example:
Request:
http colon slash slash dots2 dot DoDots dot com slash DoDots slash addShare

Response:
Standard HTTP success with response body as an xml stream in the predefined XML format of the ADDSHARE RESPONSE DTD defined earlier in this document.

getShare

The getShare method asks the server for a specific shared Dot, pack or Snapshot. The server returns the XML that was stored for sharing by the original sending user. The format of the returned XML stream is dependent on what has been "shared".

Arguments:
- usid - The unique share id for the share being requested.

Example:
Request:
http colon slash slash dots1 dot DoDots dot com slash DoDots slash getShare question mark dusid equal 1234

Response:
Standard HTTP success with response body as an xml stream in the predefined XML format. The content is returned as it was uploaded by the client.

Additional server APIs and Interfaces for Dot Providers

Dot providers can make use of all the end user APIs by using the client application. Dot Providers may also make use of additional APIs for Dot development and testing. Interfaces that call these API methods will be JSP pages. These additional APIs and JSPs will reside on the dev/test system only. Following are a number of examples of same.

addDotTemplate

The addDotTemplate method is used to add a Dot template to the database of Dot templates. This method is used by third party Dot content providers or DoDots staff.

Arguments:

- ~~dpid (optional) - A valid Dot provider id. This argument is used to associate the uploaded Dot template with a specific Dot provider. This argument is ignored if the user is not a DoDots staff member. If the user is a third-party Dot provider, the user's associate dpid stored in the database is used instead.~~

- ~~Dcid - A valid Dot category id.~~

- ~~Name - The name of the Dot. The name must be unique for the given Dot provider.~~

- ~~Desc - The extended description of the Dot.~~

~~Example:~~
~~Request:~~
~~http colon slash slash dots1 dot DoDots dot com slash DoDots slash AddDotTempalte question mark dpid equal 2 ampersand name equal scots percent 2dtest percent 2ddot ampersand desc equal The percent 2dfirst percent 2dbogus percent 2ddot percent 2dused percent 2dfor percent 2dtesting~~

~~Response:~~
~~Standard HTTP success with no response body.~~

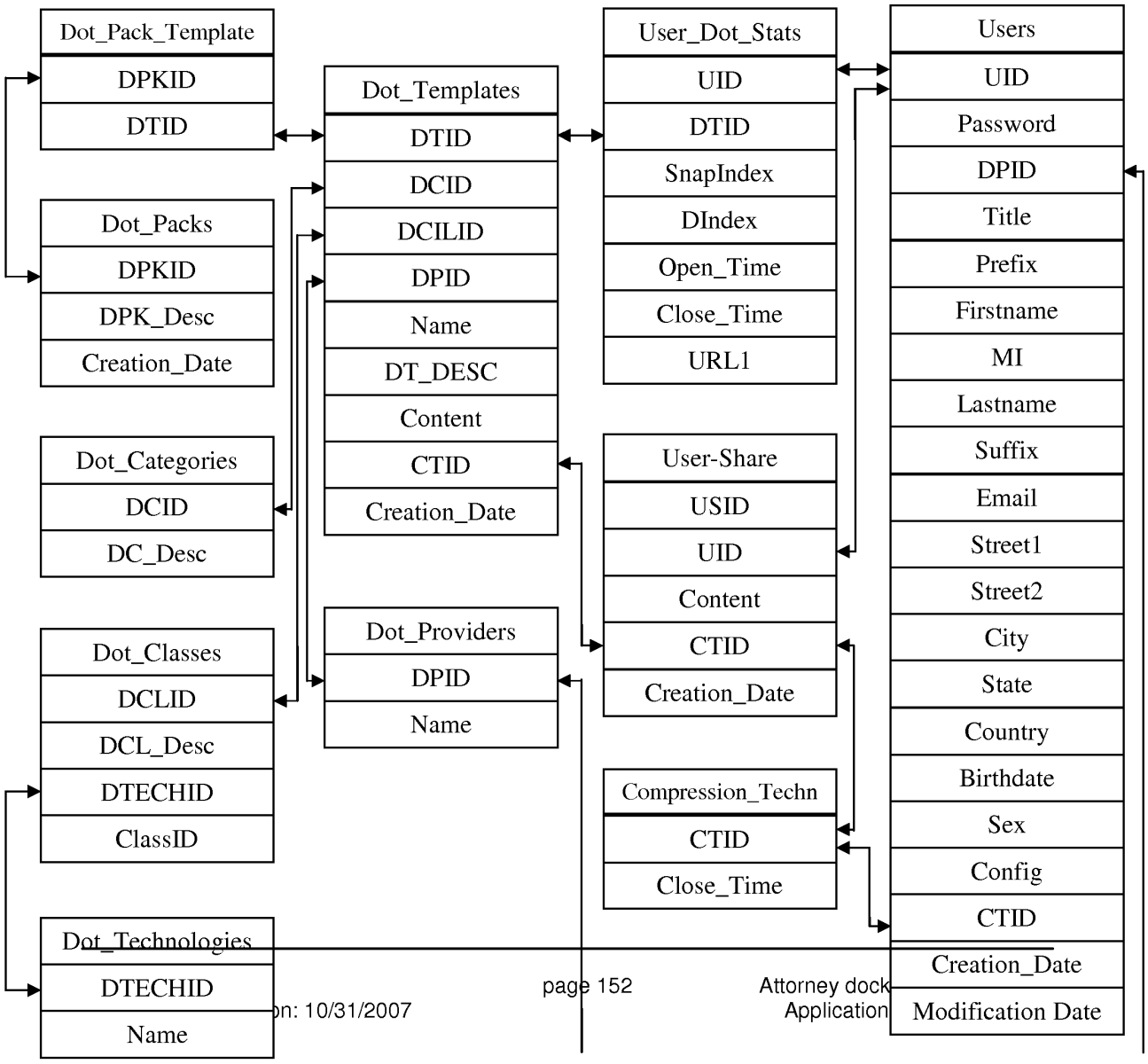
~~Additional server APIs and Interfaces for DoDots Administrator~~

~~A set of administrative functions may be provided to a DoDots administrator for high level control of the system. The DoDots administrator will have access to the end-user and Dot provider APIs as well as the admin APIs listed below. Generally, only the DoDots can make use of 100% of the server APIs. Interfaces that call these API methods will be JSP pages. These additional APIs and JSPs will reside on the dev/test system only. Due to the time-to-market constraints, in the first version of the system the interface and APIs will be the bare minimum required to allow for effective Dot system management. It is expected that more complex and complete functionality will be added to the system in subsequent releases (as loosely defined in the future sections of this document).~~

~~Database Schema~~

~~General notes on database schema and design: Usually, columns ending in "ID are unique id columns for a particular entity class. The row values for these columns may be auto-generated by the RDBMS on insert. Primary key (PK) and foreign key (FK) constraints and uniques (non-key indices) are indicated below. Primary Keys may be ordered concatenated columns in some cases as indicated by the PK Pos designation.~~

~~9.1 Entity Relationship Diagram~~



9.2 Dot_Pack_templates

Column Name	Description	Datatype	Length	Allow nulls	Unique	PK Pos	FK Table
DPKID	Dot pack ID	Smallint		No	No	1/2	Dot_pack
DTID	Dot Template	int		No	No	2/2	Dot_Templates

9.3 Dot_Packs Table

Column Name	Description	Datatype	Length	Allow nulls	Unique	PK Pos	FK Table
DPKID	Dot pack ID	Smallint		No	Yes	1/1	
DPK_Desc	Dot Pack description	nvarchar		No	No		
Creation_Da	Sate of pack creation	datetime		No	No		

9.4 Dot_Categories Table

Column Name	Description	Datatype	Length	Allow nulls	Unique	PK Pos	FK Table
DCID	Dot Category ID	Smallint		No	Yes	1/1	
DC_Desc	Dot Category Description	Nvarchar		No	No		

9.5 Dot_Classes Table

Column Name	Description	Datatype	Length	Allow nulls	Unique	PK Pos	FK Table
DCLID	Dot Class ID	Smallint		No	Yes	1/1	
DCL_Desc	Dot Class Description	Nvarchar		No	No		

DTECHID	Technology description (ie. ActiveX, Java)	nchar		No	Yes		
ClassID	Class id. (tech specific, ie. A guid or Java class spec)	nvarchar		No	Yes		

9.6 Compresion _Technologies

Column Name	Description	Datatype	Length	Allow nulls	Unique	PK Pos	FK Table
DTECHID	Compression	Smallint		No	Yes	1/1	
Name	Compression Technology Name	Nvarchar		No	Yes		

9.7 Dot_Technologies

Column Name	Description	Datatype	Length	Allow nulls	Unique	PK Pos	FK Table
DTECHID	Dot Technology ID	Smallint		No	Yes	1/1	
Name	Dot Technology Name	Nvarchar		No	Yes		

9.8 Dot_Template Table

Column Name	Description	Datatype	Length	Allow nulls	Unique	PK Pos	FK Table
DTID	Dot Template ID	int		No	Yes	1/1	
DCID	Dot Category ID	Smallint		No	No		Dot_categories
DCILID	Dot Class ID	Smallint		No	No		Dot_Classes
DPID	DotProvider ID	Smallint		No	No		Dot_Providers

Name	Name of the Dot	nvarchar		No	No		
DT_DESC	Dot Description	nvarchar		No	No		
Content	Dot Template Content	Varbinary		No	No		
Content_ Compression CTID	Type of compression the content is stored in	Nvarchar		No	No		

9.9 Dot_Providers Table

Column Name	Description	Datatype	Length	Allow nulls	Unique	PK Pos	FK Table
DPID	Dot provider ID	Smallint		No	Yes	1/1	
Name	Dot Provider Name	Nvarchar		No	Yes		

9.10 Dot_User_Events Table

Column Name	Description	Datatype	Length	Allow Nulls	Unique	PK Pos	FK Table
DUID	UserID	Int		No	No		Dot_Users
DEID	Dot event ID	Int		No	No		Dot_Events
DTID	Dot template ID	Int		Yes	No		Dot_Templates
DPID	Dot Provider ID	Int		Yes	No		Dot_Providers
Start_Time	Start Time info for Dot event in UTC/ GMT time	Datetime		No	No		
End_Time	End Time info for Dot event in UTC/GMT time	Datetime		Yes	No		
URL1	URL info 1 for Dot event (fully qualified) for events that involve URLs	String		Yes	No		

9.11 Dot_Events Table

Column Name	Description	Datatype	Length	Allow Nulls	Unique	PK Pos	FK Table
DEID	Dot Event ID	Int		No	Yes	1	
DE_Name	Dot Event Name	String		No	Yes		
DE_Desc	Dot Event Description	String		No	Yes		

9.12 Dot_User_Shares Table

Column Name	Description	Datatype	Length	Allow Nulls	Unique	PK Pos	FK Table
DUSID	UserShare ID	Int		No	Yes	1/1	
DUID	User ID	Int		No	No		Users
DUS_Content	The content of the shared item. This could be a dot, Dot pack, or Snapshot	Varbinary		No	No		
DCTID	The type of compression used to store to share	Number	9	No	No		
DUS_Creation_Date	Date of creation of the share	Datetime		No	No		
DUS Download_ Date		Datetime		No	No		

9.13 Users Table

Column Name	Description	Datatype	Length	Allow Nulls	Unique	PK Pos	FK Table
Username	User ID	Nvarchar		No	Yes	1/1	
Password	User's Password	Nvarchar		No	No		

DPID	Dot Provider ID (only in dev/test and only if user is a Dot provider developer)	Smallint		Yes	No		Dot_Providers
Title	Business	Nvarchar		Yes	No		
	Title, if applicable						
Prefix	Prefix to name (i.e., Mr., Mrs., Dr., etc.)	Nchar	10	Yes	No		
Firstname	First Name	Nvarchar	25	No	No		

MI	Middle Initial	Nvarchar	1	Yes	No		
Lastname	Last Name	Nchar	25	No	No		
Suffix	Name Suffix (i.e. Jr., Sr., III, PhD.)	Nchar	10	Yes	No		
Email	Email Address	Nvarchar		No	No		
Street1	First line of street address	Nvarchar		Yes	No		
Street2	Second line of street address	Nvarchar		Yes	No		
City	City	Nvarchar		Yes	No		
State	State	Nvarchar		Yes	No		
Country	Country	Nvarchar		Yes	No		
Birthdate	Birthdate	Datetime		Yes	No		
Sex	Sex (m, f, n)	Nchar		Yes	No		
Config	Last uploaded configuration dump	Nvarchar		No	No		
Config_Compression	Compression format of the config content	Nchar	10	No	No		
Creation_Date	Date of creation of the	Datetime		No	No		
Modification_Date	Date of last activity	Datetime		No	No		

BACKGROUND OF THE INVENTION

(03) A user operating a client computer typically accesses the Internet by using a viewer application, such as a browser to view web content provided at a destination address, typically a web page. In this context, web content and web applications are designed to fill the entire web page. It is known to divide the web content into different regions of a single web page. For example, personalized web pages can be specified, such that a user views a variety of content sources in a single page, such as stock information, weather information, and sports information, which is aggregated at the server that delivers the web page to the user, who then views the aggregated content in a single web page. Observe that even when disparate content is aggregated, in this manner, it is reassembled into a full web page and is served through a full-screen browser. Web content and application developers therefore have limited control over the user experience: content is typically trapped within the frame of the browser. A developer's only alternative to engaging a user page-by-page in a browser is to develop, distribute, and support custom client software. In the Web browser scenario, it is the content provider, not the user that aggregates the information that is viewed by the user. Thus, the user is not in a position to separately aggregate the content at a client computer, instead the user is constrained to view the content that has been delivered in the manner provided by the server computer hosting the web page. There is a growing desire for individual users to fully control the aggregation and presentation of content and web applications that appears on a client computer.

(04) A user who wishes to view multiple web pages or applications can open multiple instances of a browser. However, the user will not be able to view each "full-screen" page at the same time. Instead, the user must adjust the windows corresponding to each browser

instance and view only part of each page. The information appearing in each browser is not designed for viewing in this manner. Thus, the user cannot create an optimized display of content from multiple sources.

(05) Currently, content providers and end users have limited tools to alter the browser in which content appears. That is, the controls associated with a browser are not fully configurable. Thus, the vendor of a browser is in a position to brand the browser and regulate the controls associated with the browser. There is a growing desire for content providers to not only fill a browser with their content, but to also fully brand and control the frame in which the content appears. Further, in some instances, content providers desire to limit the controls associated with a browser or viewer, so that a user is more inclined to view a single set of content, for example, by having limited access to previously viewed content.

(06) At the present time, it is relatively difficult to trace the content viewing activity of a client computer. In other words, it is difficult to identify the type of content that a particular user of a client computer favors. Consequently, there are limited tools 25 available to provide a user with tailored information that would be of particular interest to the user.

(07) In view of the foregoing, there is a need in the art to provide a technique for accessing multiple instances of distributable computer readable media in their entirety simultaneously, where these instances are typically smaller than the full pages used in 30 current web pages and web applications. There is a further need for providing the user with flexibility in selecting, collecting, relating and viewing such computer readable media, and for giving the media provider flexibility in directing media to a specific user and controlling the framework in which media is presented. Finally, there is a need to gather more accurate

information regarding the type of content that a user enjoys, so that the user can be automatically provided with this content.

SUMMARY OF THE INVENTION

(08) The invention includes a method of presenting distributable computer readable media to a user in response to a user request. The method comprises the steps of identifying a definition of a Networked Information Monitor (NIM). A NIM frame is defined for the NIM using the definition. Content is then retrieved for the NIM. Then, the content is placed in a NIM viewer defined by the frame.

(09) The invention also includes a method of altering a Networked Information Monitor (NIM). The method includes the step of receiving a message at a NIM. The message specifies a configurable feature of the NUM. The NIM is altered in accordance with the configurable feature of the message.

(10) The apparatus of the invention includes a computer readable memory to direct a computer to function in a specific manner. The computer readable memory includes a first executable module to identify a definition of a Networked Information Monitor (NIM). A second executable module defines a NIM frame for the NIM using the definition. A third executable module retrieves content for the NIM. A fourth executable module places the content in a NUM viewer defined by the frame.

(11) The apparatus of the invention further includes a computer readable memory with a first executable module to receive a Networked Information Monitor (NIM) message. The NIM message specifies a configurable feature of a NIM. A second executable module alters the NIM in accordance with the configurable feature of the NIM message.

BRIEF DESCRIPTION OF THE DRAWINGS

(12) FIG. 1 illustrates a system for processing distributable computer readable media in accordance with one embodiment of the present invention;

(13) FIG. 2 shows exemplary processing steps used to initiate an instance of a home networked information monitor (NIM) using the apparatus of FIG. 1;

(14) FIG. 3A illustrates a screen logo in accordance with one embodiment of the present invention;

(15) FIG. 3B illustrates a login construct in accordance with one embodiment of the present invention;

(16) FIG. 4 illustrates a set of visual manifestations corresponding to a collection of NIMs, including a home NIM;

(17) FIG. 5 illustrates a visual manifestation of a home NIM as well as a visual manifestation of a reference NIM that references additional NIMs;

(18) FIG. 6 illustrates how a NIM, referenced by the reference NIM of FIG. 5, is added to a processed user profile in response to a designated keyboard entry sequence or mouse click;

(19) FIG. 7 illustrates how a NIM is shared with other users in accordance with an embodiment of the present invention;

(20) FIGS. 8A and 8B illustrate screen panels that facilitate the collection of the description of a set of designated NIMs into a pack;

(21) FIG. 9A illustrates how the relative position of visual manifestations that correspond to NIMs remains fixed when the visual manifestations are within a predetermined distance of each other;

(22) FIG. 9B illustrates a representative pack in accordance with the present 20 invention;

(23) FIGS. 10A, 10B and 10C illustrate how a set of visual manifestations corresponding to a collection of NIMs is aligned against a boundary when a user selects the visual manifestations and pushes them against the boundary;

(24) FIG. 11 is a flow chart of the steps taken by a client to give a user access to a NIM where the user "collects" the NIM, in accordance with one embodiment of the invention;

(25) FIG. 12 is a flow chart of the steps taken by a client to present a NIM to a user, where the NIM has been "collected" previously by the user, in accordance with one embodiment of the invention;

(26) FIG. 13 illustrates a data structure for a NIM definition, stored in the NIM application server's template database or user profile database;

(27) FIG. 14 illustrates NMA message routing between NIMs and the message interface in the client parser application;

(28) FIG. 15 is a diagrammatic illustration of an embodiment of a NIM Management Module utilized in accordance with an embodiment of the invention;

(29) FIG. 16 is a diagrammatic illustration of an embodiment of the NIM Templates database utilized in accordance with an embodiment of the invention;

(30) FIG. 17 is an illustration of a main NIMIndex Web page used in accordance with an embodiment of the invention;

(31) FIG. 18 is an illustration of a single NIMIndex category used in accordance with an embodiment of the invention;

(32) FIG. 19 is an illustration of a full description of NIM content provided in accordance with an embodiment of the invention;

(33) FIG. 20 is an illustration of a Web page displayed to the user once the user has clicked to collect the NIM;

(34) FIG. 21 is an illustration of the main home NIM graphical user interface used in accordance with an embodiment of the invention;

(35) FIG. 22 is an illustration of a "Get New NIM" graphical user interface that may be used in accordance with an embodiment of the invention;

(36) FIG. 23 is an illustration of a "More NIMs" graphical user interface representative of an embodiment of the invention;

(37) FIG. 24 is a diagrammatic illustration of the ShareLink database used in accordance with an embodiment of the invention;

(38) FIG. 25 is an illustration of a Share NIM's graphical user interface according to an embodiment of the invention;

(39) FIG. 26 is an illustration of the main DevZone Web page utilized in accordance with an embodiment of the invention;

(40) FIG. 27 is a partial view of a NIM modification web page utilized in accordance with an embodiment of the invention;

(41) FIGS. 28A to 28D are graphical user interfaces of development NIMs (DevNIMs) utilized in accordance with an embodiment of the invention;

(42) FIGS. 29A and 29B are illustrations of Administrative Zone (AdminZone) Web pages utilized in accordance with an embodiment of the invention;

(43) FIGS. 30A and 30B are also illustrations of Administrative Zone (AdminZone) Web pages utilized in accordance with an embodiment of the invention;

(44) FIGS. 31A and 31B are further illustrations of Administrative Zone (AdminZone) Web pages utilized in accordance with an embodiment of the invention;

(45) FIG. 32 illustrates an embodiment of an event log that may be used in accordance with an embodiment of the invention;

(46) FIG. 33 illustrates the tracking of events in an event log module in accordance with an embodiment of the invention;

(47) FIG. 34 illustrates a statistics database that may be used in accordance with an embodiment of the invention;

(48) FIG. 35 illustrates a statistical analysis module and a content analysis module that may be used in accordance with an embodiment of the invention;

(49) FIG. 36 illustrates a content database that may be used in accordance with an embodiment of the invention; and

(50) FIG. 37 illustrates a user account database that may be used in accordance with an embodiment of the invention.

DETAILED DESCRIPTION OF THE INVENTION

(51) The present invention discloses a technology that is capable of processing distributable computer readable media. Distributable computer readable media includes, but is not limited to, standard web content, such as HTML, dHTML, images, imbedded ActiveX and Java applications, JavaScript, CSS, Pen scripts, Streaming Media, and/or Flash. The present invention is advantageous relative to prior art systems and methods because it provides improved mechanisms for simultaneously interacting with several independent sources of distributable computer readable media, collecting references to such media, and

sharing such references with other users. The ac disclosed technology is further advantageous because it provides improved systems and methods for on screen management of distributable computer readable media.

(52) In the system and method of the present invention, a user logs into a server by providing a login identifier to a login construct. The login identifier is used by the server to obtain an unprocessed user profile that corresponds to the user. The unprocessed user profile is processed by the server to generate a processed user profile. Advantageously, this processing step allows for up-to-date refinement of the user profile. Up-to-date refinements include, for example, the addition of advertisements directed to the user based on one or more characteristics in the user profile. The processed user profile is delivered to the client computer associated with the user. The processed user profile includes references to the networked information monitors (NIMs). As used herein, the term networked information monitor or NIM refers to a fully configurable frame with one or more controls; the frame through which content is optionally presented. The fully configurable frame utilized in accordance with the invention stands in contrast to present web browsers, which are branded by the browser vendor and which have limited means by which to alter the controls associated with the browser.

(53) Attention is initially directed toward the home NIM of the invention, which coordinates the activities of all other NIMs that are accessed by a user. The home NIM facilitates much of the technology of the present invention, including the ability to simultaneously review multiple sources of distributable computer readable media as well as to package and distribute such media.

(54) FIG. 1 is a general illustration of a system in accordance with one embodiment of the present invention. In FIG. 1, a network 10 is operated in accordance with the present invention. Network 10 includes at least one user or client computer 20, at least one server computer of class 50, and optionally one or more server computers of class 82. User computer 20 as well as server computers of class 20 50 and 82 are each connected by transmission channel 44, which is any wired or wireless transmission channel.

(55) User computer 20 is any device that includes a Central Processing Unit (CPU) 24 connected to a random access memory 30, a network connection 28, and one or more user input/output ("i/o") devices 40 including output means 42. Output means 42 is any device capable of communicating with a user and includes, for example, a video monitor, a liquid crystal display, voice user interfaces, and/or integrated graphic means such as mini-displays present in web-phones. Typically, user computer 20 includes a main non-volatile storage unit 22, preferably a hard disk drive, for storing software and data. Further, user computer 20 includes one or more internal buses 26 for interconnecting the aforementioned elements. In a typical embodiment, memory 30 includes an operating system 32 for managing files and programs associated with user computer 20. In some embodiments, operating system 32 includes a registry 34 that has one or more references to specified locations in system 10. The exemplary memory 30 of FIG. 1 further includes a web browser 36 for viewing web content and a client parser application 38 for facilitating low level functionality, such as login and logout procedures, in accordance with the present invention. In some embodiments, client parser application 38 uses the one or more references in registry 34 to obtain a login construct from server 50. In various embodiments, in accordance with the present invention, client parser application 38 runs in

conjunction with one or more software modules, such as an event log module 98, which tracks user activity, a message interface module 106, which serves as a communication interface between the client parser application 38 and web server 58 and/or external web servers, a home NIM 108, which references one or more NIMs 110, and a visual management system 114 which regulates the characteristics of visual manifestations of NIMs 108 and 110 when displayed on output device 42. Furthermore, in some embodiments, client parser application 38 runs in conjunction with instances of web browser 36 as well as web server 58 as detailed below.

(56) Server computer 50 includes standard server components, including a network connection device 46, a CPU 52, a main non-volatile storage unit 54, and a random access memory 56. Further, server computer 50 includes one or more internal buses 48 for interconnecting the aforementioned elements. Memory 56 stores a set of computer programs, modules and data to implement the processing associated with the present invention.

(57) The embodiment of memory 56 illustrated in FIG. 1 includes a web server 58 for processing requests received from client computer 20. Web server 58 has many components, including a variety of modules and data structures to assist users that want to log into system 10. Namely, login module 60 handles an entry request from a client computer 20 and accepts a login identifier that corresponds to a user from client computer 20. Login constructor 62 generates a login construct in response to a call for a login construct and transfers the login construct to client 20. Login constructor 62 dynamically generates login constructs using updated login NIM content 64. Login validation module 66 works in conjunction with user profile database manager 100 to identify an unprocessed

user profile, which is associated with a user provided login, in user profile database 76. If an unprocessed user profile corresponding to the user provided login does not exist in user profile database 76, login validation module 66 associates a new unprocessed user profile with the login identifier.

(58) When an unprocessed user profile is identified by login validation module 66, it is processed by user profile processor module 68 to produce a processed profile. It will be appreciated that the services of user profile processor module 68 are highly advantageous because they allow for last minute user profile revisions. Such revisions include, for example, the addition or customization of NIMs referenced by the user profile, and/or server redirect information that is a function of current server load in system 10. Importantly, the processed user profile includes a reference to a home NIM. The home NIM is capable of accessing each of the NIMs that are represented in the processed user profile. Memory 56 further includes delivery module 70, which coordinates the delivery of portions of the home NIM to the client based on parameters specified in the processed user profile.

(59) Once a user has successfully logged into system 10, request server module 72 handles requests for specified NIMs from client 20. When such a request is received, request server module 72 routes the request to an address that corresponds to the specified NIM and transmits the specified NIM to client 20. One class of specified networked information handled by request server module 72 is requests for NIMs. When such a request is received, request server module 72 searches NIM templates database 74 for the specified NIM. NIM templates database 74 includes a large number of NIM templates. Each NIM template defines the characteristics of a specific NIM, including fully configurable frame characteristics, viewer and control characteristics, and NIM content references.

(60) The web server 58 illustrated in FIG. 1 further includes additional modules 102 to handle specialized features of the present invention. For example, one embodiment of the present invention provides a mechanism that allows users to distribute NIMs to each other. In such embodiments, a special server module 102 provides instructions for storing the NIMs, which are to be distributed, in sharelink database 78. Advantageously, NIMs that are distributed to other users are customizable. A user can, for example, resize and position a particular NIM prior to sharing it with another user. Indeed, it is possible, in such embodiments, for a user to arrange a series of NIMs in a unique arrangement and then distribute the collection of NIMs in the designated NIMs in the designed arrangement. As an illustration, a user arranges a first NIM that represents a scrolling stock ticker at the bottom of an output means, such as a computer screen, a second NIM that tracks the NASDAQ top ten most heavily traded stocks in the upper left corner of the output means, and a third NIM that tracks headline news on the upper right hand corner of the output means. Then, the user distributes the three NIMs in this customized arrangement to other users. Observe that in this example a user of a client computer is aggregating separate sets of information in different NIMs. This stands in contrast to prior art approaches where a web server running on a server computer aggregates information in a single page.

(61) System 10 is highly scalable and thus supports a large number of users. This scalability stems from the fact that the server 50 is delivering the definition associated with a NIM. The content displayed in the NIM may be located on a separate computer.

(62) Memory 56 provides a statistical analysis module 104 for tracking key events associated with users. This information is stored in statistics database 80. The information collected by statistical analysis module 104 is used for a wide variety of purposes,

including server load optimization and directed advertising, as discussed below. As described below, the statistical information gathered in accordance with the invention includes fully traced events defining the type of content and the duration over which all content is viewed by a user. This type of comprehensive information is not available using present techniques.

(63) Much of the distributable computer readable media that is available for processing is stored as content elements 94 on server 82. Server 82 is a standard web server that includes components such as a network connection device 88, a CPU 86, a main non-volatile storage unit 84, a random access memory (RAM) 92, and one or more internal buses 90 for interconnecting the aforementioned elements. RAM 92 includes some of the content elements 94 stored by server 82. Other content elements 94 are stored in storage unit 84. In some embodiments, a single web server 58 is capable of directly accessing content elements 94 located on one or more servers 82. In other embodiments, each server 82 has a resident web server module that works in conjunction with server 50 to identify, optionally dynamically generate, and serve content elements 94 upon demand.

(64) Now that general architecture of a system in ac dance with the present invention has been disclosed, attention turns to FIG. 2, which discloses a method for logging into system 10 (FIG. 1). In the first processing step shown in FIG. 2 (202), a user initiates a session on system 10 by requesting the global login script "session_config." It will be appreciated that the term "session_config" merely provides an illustrative name for the global login script and that the technology of the present invention is by no means limited to this name or the script described.

(65) The request for "session_config" originates on client 20 and is sent to server 50 where it is processed by login module 60 of web server 58. Upon receiving request 202, login module 60 creates a "session_config" global login script (204). Processing step 204 is advantageous relative to systems that have static global login scripts because it allows for the incorporation of highly variable information. This highly variable information includes, for example, system settings such as up-to-date server redirect information, server content address changes, directed advertisements, and messages. An exemplary "session_config" is found in Example 1 below. Each line of data has an associated numeral. The remaining text in the example describes the purpose of select data.

(66) EXAMPLE 1

(67) Version tag that identifies the latest home NIM version

(1) </SESSION_CONFIG VERSION="alpha:3"

(68) Upgrade event that is sourced when home NIM version is outdated

(2) <UPGRADE=http://www.NIM.com /QuickOpen.exe>

(69) LOCATION OF SERVER-SUPPORTED FUNCTIONALITY Default base address for server supported functionality

(3) <METHODS BASEURL=http://neo.NIM.com /servlet/NIMServer/>

(4) <ADD_USER URL="addUser/>

(5) <GET_USER URL="getUser"/>

(6) <SET_USER URL="setUser/>

(7) <GET_SES_CONFIG URL=http://www.NIM.com /home_NIM/s_cnfg.xml"/>

(70) Server-based functionality for setting password

(8) <SET_PASSWORD URL="setPassword"/>

(71) List of all publically available NIMs

(9) <GET_MASTER_NIM_LIST URL="getMasterNIMList"/>

(10) <GET_ALL_CONFIG URL="getAllConfig"/>

(11) <SET_ALL_CONFIG URL="setAllConfig/>

(12) <SET_ALL_STATS URL="setAllStats"/>

(13) <GET_NIM_TEMPLATE URL="get_NIM_Template"/>

(72) Location of server-side NIM and pack sharing functionality

(14) <ADD_SHARE URL="addShare"/>

(15) <GET_SHARE URL="getShare"/>

(16) <AUTH_TEST URL="DOeCHO?AUTH=TRUE"/>

(73) Redirect information

(17) <DO_REDIR URL="doRedir"/>

(18) </METHODS>

(74) Flexible content layer that defines default NIM frame appearance, including the default appearance of the frame of a home NIM

(19) <FRAMES>

(75) Default NIM frame appearance

(20) <NIM>

(21) <IMAGES BASEURL=

http://www.NIM.com /home_NIM/NIM_FrameImages/>

(22) </NIM>

(76) Default home NIM frame appearance

(23) <HOME_NIM>

(24) <IMAGES BASEURL=

http://www.NIM.com /home_NIMImages/>

(25) </HOME_NIM>

(26) </FRAMES>

(77) Location of system NIM templates

(27) <NIMs>

(28) <ADD TEMPLATE="http://www.NIM.com / . . . /add_NIM_XML.xml"/>

(29) <HELP TEMPLATE="http://www.NIM.com / . . . /help_NIM_XML.xml"/

(30) <LOGIN TEMPLATE="http://www.NIM.com / . . . /login2.xml"/

(31) </NIMs>

(32) </SESSION_CONFIG>

(78) Line 1 of the exemplary "session_config" of Example 1 provides the version tag for the expected version of the home dot system that corresponds to the "session_config" script. In one embodiment, client parser application 38 determines whether it is up-to-date using the information in line 1. If client parser application 38 determines that it is outdated, an upgrade request is made in accordance with the instructions provided by the UPGRADE flag of line 2. In one embodiment, the UPGRADE flag in line 2 of Example 1 describes the location of an executable program, one of skill in the art will appreciate that this flag can in fact reference any form of instruction, including a flat file, a web page, a script, a symbol, or an address.

(79) Lines 3 through 18 in Example 1 define the functionality that is provided by a server, such as server 50. For example, line 8 of Example 1 provides the location of a set of

instructions that are called when a user requests a password change. Furthermore, line 9 of Example 1 provides the location of master list of NIMs that are publically available. Lines 14 and 15 of Example 1 provide the location of specialized server-side functionality that allows users to share data such as NIM definitions.

(80) Lines 19 through 26 of Example 1 define where the default appearance of a NIM and a home NIM are found within system 10. Lines 27 through 31 define a collection of system NIMs. A system NIM is any type of NIM that is to be distributed to each user of system 10. In some embodiments, system NIMs are used to provide a core functionality. In Example 1, line 28 defines a NIM that provides users with a convenient mechanism for collecting additional NIMs. Line 29 defines the location of a NIM that is invoked when the user presses a help button associated with a home NIM. Finally, line 30 defines the location of a NIM that is used to log into system 10.

(81) Returning to FIG. 2, once login module 60 has created "session_config," it is sent back to requesting client 20 (206). When a "session_config" is received by client 20, client parser application 38 parses the global login script in order to identify a reference to a login constructor 62 (208). Login constructor 62 is a server-based module that generates a construct that allows a user to log into system 10. When client parser application 38 locates the reference to login constructor 62 in "session_config," a request for a login construct is directed to the identified reference (210). In Example 1 above, the reference to the login construct is provided in line 30. On line 30, the global variable "LOGIN TEMPLATE" is assigned the URL address "http://www.NIM.com . . . /login2.xml." Client parser application uses the URL assigned to the global variable "LOGIN TEMPLATE" to make a

request for a login constructor 62 that is directed to this URL. When login constructor 62 receives a request for a login construct, it generates a login construct (212).

(82) Login construct 148 (FIG. 3B) illustrates a type of login construct that is generated in one embodiment of the present invention during processing step 212. Before the login construct is executed on client 20, a schematic such as logo 146 (FIG. 3A) is displayed on output means 42. As illustrated in FIG. 1, login constructor 62 is a component of web server 58. However, there is no requirement that login constructor 62 be a component of web server 58. In fact, login constructor 62 is a standalone software program in some embodiments of the present invention whereas in other embodiments, login constructor 62 is merely a script, such as a PERL script, that is processed by an interpreter program native to server 50. In still other embodiments, login constructor 62 is merely a simple flat file that includes a set of instructions that are interpretable by client parser application 38. In such embodiments, login constructor 62 is the login construct. In embodiments in which a login construct is dynamically generated, it is possible to introduce last minute changes in the login construct. Thus, an advantage of the exemplary login process shown in FIG. 2 is that there are multiple stages in which updated information is used to customize the login process based on the environmental variables.

(83) Once a login construct has been prepared by login constructor 62, it is transferred back to client 20 (214) (FIG. 2) and executed in conjunction with client parser application 38 (216). The login constructs of the present invention are a form of NIM. Therefore, one function of processing step 216 is to obtain the login NIM content 64 (FIG. 1) specified by the login construct from server 64. In login construct 148, for example, the login NIM content includes the shape and functionality of "Exit button 160," message 150, the shape

and functionality of "New user" button 152, the functionality of "Forgot it?" button 154, and login panel 156. When processing step 216 is completed, the user uses the login construct to provide a login identifier (218).

(84) In FIG. 3B, a user has provided the login identifier "Galliani." The definition of login identifier as used in the present invention is to be broadly construed. In some embodiments, login identifiers include a unique name and a corresponding password. In other embodiments, a login identifier does not have a password. This is particularly the case when the user is a guest or a new user and there is no user profile associated with the user.

(85) Working in conjunction with client parser application 38, the login construct accepts the user login and sends it to server 50 for validation (218). As illustrated in the exemplary system of FIG. 1, web server 58 includes a login validation module 66 to verify the login identifier provided by user (220). Typically, processing step 220 involves a look-up operation in which the login identifier is used to query user profile database 76 for an unprocessed or raw user profile that matches the login identifier. In embodiments that include a password, validation step 220 includes a password verification step. Successful completion of processing step requires entry of a valid login identifier sequence in processing step 218. When processing step 220 has been successfully completed, the raw or unprocessed user profile corresponding to the login identifier is obtained from user profile database 76 (FIG. 1) (222) and is processed by user profile process module 68 to produce a processed or finalized user profile that is delivered to client 26 (226). In sonic embodiments, a user profile 76 includes user contact information, such as the name, address, telephone number and email address of a user. Additionally, some embodiments of system 10 provide different types of access privileges. For example one embodiment of the

present invention includes developer access privileges, administration access privilege, and general user access privileges. In such embodiments, the access privileges that have been granted to a user are stored in the user profile 76 associated with the user.

(86) The processed user profile includes a reference to each NIM in system 10 that is associated with the login identifier provided in processing step 216. One of the NIMs referenced by the processed user profile is the home NIM that corresponds to the login identifier provided in processing step 216. When executed in conjunction with client parser application 38 in processing step 226, the home NIM provides a mechanism for accessing each of the NIMs referenced by the processed user profile. Like the login construct, the home NIM includes several components, including pull down menus and screen manipulation functionality. The reference to the home NIM in the processed user profile includes the system 10 address of each of these components. Therefore, in one embodiment, construction of the home NIM in processing step 226 involves one or more requests to server 50 and/or server 82 for content (228) that is then rendered (230) in accordance with the home NIM description provided in the processed user profile. In some embodiments, the home NIM is distinct from other NIMs in the sense that a large proportion of the home NIM in such embodiments is pre-compiled. Such embodiments are advantageous because some of the functionality provided by the home NIM requires substantial client 30 processing resources. Therefore, to minimize such processing resource requirements, many aspects of the home NIM are pre-compiled in some embodiments. In other embodiments, however, the home NIM has a structure that is substantially the same as a regular NIM. In such embodiments, simple script commands are used to identify the NIM as a home NIM.

(87) Upon completion of processing step 230, the user is granted access to all of the technologies of the present invention, including the ability to view multiple NIMs simultaneously, collect new NIMs, customize NIMs, and share customized NIMs with other users. An exemplary processed user profile is provided in Example 2. Once again, each line of data is identified with a numeral, while the remaining text in the example describes select data. In some embodiments, the user is granted specific privileges and the extent to which the user is granted access to system 10 is regulated by the types of privileges that have been granted to the user.

(88) EXAMPLE 2

(1) SAMPLE PROCESSED USER PROFILE

(2) <ALL CONFIG>

(89) NIMs AND PACKS THAT CORRESPOND TO THE USER

(3) USER

(90) NIM definition 1

(4) <NIM DOMAIN="ZDNet" GLOBALID="1" KND="news"

(5) <FRAME BACKGROUNDCOLOR=#FFFF00" COLLAPSED="FALSE"

(6) FIXHEIGHT="TRUE" FIXWIDTH="TRUE" NAME="ZDNet Breaking News"

(7) PIXELHEIGHT="275" PIXELWIDTH="235" X="RIGHT" Y="TOP">

(8) <TITLE COLOR=#000000" JUSTIFY="RIGHT" TEXT=" "/>

(9) <TITLEBARIMAGE DOWN=http://www.NIM.com / . . . /feed/titlebar.gif

(10) HOVER=URL address to a first GIF file <param 1> . . . <param N>

(11) INACTIVE=URL address to a second GIF file <param 1> . . . <param 2>

(12) NORMAL=URL address to a third GIF file <param 1> . . . <param 2>

(13) <BOTTOMBARIMAGE DOWN=

http://www.NIM.com / . . . /feed/bottombar.gif

(15) HOVER=URL address to a fourth GIF file <param 1> . . . <param 2>

(16) INACTIVE=URL address to a fifth GIF file <param 1> . . . <param 2>

(17) NORMAL=URL address to a sixth GIF file <param 1> . . . <param 2>

(18) </FRAME>

(19) <MENU/>

(20) <CONTROL_LAYOUT HEIGHT="1" HEIGHTSCALES="TRUE" WIDTH="1"

(21) WIDTHSCALES="TRUE"> <CONTROL CLASS="Browser" HEIGHT="1"

(22) ID="1" KIND="A" LEFT="0" TOP="0"

(23) URL=http://www.mandala.com /cgl/zdnet/zdfeedl.cgi WIDTH="1"/>

(24) </CONTROL_LAYOUT>

(25) <CATEGORIES/>

(26) <EVENTS/>

(27) </NIM>

(91) NIM definition 2

(28) <NIM DOMAIN=NIM DOMAIN 2 GLOBALID=`2`

(29) </NIM>

(92) NIM definition N

(30) <NIM DOMAIN=NIM DOMAIN 2 GLOBALID=`N`

(31) </NIM>

(93) Pack definition 1

(32) <PRESET TITLE="New DotPack">

(33) <NIM GLOBALID="1" X="RIGHT" Y="TOP"/>

(34) <NIM GLOBALID="2" X=RIGHT Y="320"/>

(35) </PRESET>

(36) </SHARE>

(94) Last state of the home NIM

(37) <LASTSTATE>

(38) <PRESET TITLE=" ">

(39) <NIM GLOBALID="1" X="RIGHT" Y="TOP"/>

(40) <NIM GLOBALID="2" X=RIGHT Y="280"/>

(41) </PRESET>

(42) <HOMENIM COLLAPSED="FALSE" HEIGHT="134" X=616" Y="109"/>

(43) </LASTSTATE>

(44) </ALL_CONFIG>

(95) Example 2 describes a representative processed user profile in accordance with the present invention. In general, a processed user profile includes three major components: (i) a definition of each NIM associated with the user, (ii) a description of each pack associated with the user and, (iii) the last state of each home NIM associated with a user. In Example 2, the definition of each NIM associated with the user is found on lines 4 through 31. Specifically, lines 4 through 31 describe NIM definitions I through N. In Example 2 there is only one pack associated with the user. This pack, entitled "New DotPack," is found on lines 32 through 35 of Example 2. The final major component of the processed user profile found in Example 2 is the last state of the home NIM, which is defined on lines 37 through 43. This code stores the last state of the home NIM. Such last state information includes whether the home NIM was collapsed, and the position of the home NIM on the screen.

(96) When the user wishes to log out of system 10, the processed user profile is transferred from client 20 to server 50. When web server 58 receives the processed user profile, it passes the processed user profile to user profile database manager 100. User profile database manager 100 stores the processed user profile as the unprocessed user profile 76 corresponding to the user. In some embodiments, such a storage operation involves a conversion process. For example, advertisements or specific system NIM definitions are stripped from the processed user profile in order to convert the processed user profile to the

unprocessed user profile 76 that corresponds to the user. In some embodiments, the processed user profile is periodically transferred, in its entirety or incrementally, from client 20 to server 50 and saved in the manner described in the log out procedure above. Such timed periodic or event based backup procedures are possible because NIM definitions are efficiently described, thus the absolute size of a processed user profile remains relatively small. Accordingly, timed backups of a processed user profile to user profile database 76 are possible without extensive use of system 10 bandwidth or server 50 resources.

(97) At this stage, a system (FIG. 1) and a login procedure (FIG. 2) in accordance with the present invention has been disclosed. Although the system and login procedure was discussed using an example where only one home NIM was associated with a user, it will be appreciated that, in some embodiments, any number of distinctly different home NIMs are associated with a user. Furthermore, a user can simultaneously execute multiple instances of a particular home NIM on client 20 or, indeed, any number of different home NIMs. In one embodiment, a developer or merchant provides a user with a highly customized home NIM that provides specialized functionality. In such embodiments, the user collects the home NIMs and, therefore, a processed user profile includes a description of more than one home NIM.

(98) Attention now turns to some of the advantages and features of the present invention. In FIG. 4, a visual manifestation of the home NIM 162 is illustrated. One advantage of the home NIM, which is an advantage that is common to NIMs in general, is that the content of the NIM is not trapped in a third party viewer. In fact, the home NIM definition regulates the actual appearance of the home NIM. The home NIM definition is formed by general

parameters and commands found in "session_config" as well as customized parameters and commands in the processed user profile. The division of the home NIM definition between a system level file and a user level file represents a balance in the tension between the need for a system 10 host to insure a consistent level of quality, through the proper implementation of general parameters and commands, and the desire of each user to create highly customized home NIMs. Lines 20 through 22 of Example 1 provide an example of general parameters that are defined in "session_config." Lines 23 through 25 define the source location of home NIM frame images. In home NIM 162 (FIG. 4), such home NIM frame images include the image used to represent buttons 164 through 174, and menu tabs 1 d 178. Furthermore, lines 20 through 22 of Example 1 define the location of other images that are used to construct default NIMs. An example of user initiated home NIM customization is found in lines 33 through 39 of Example 2, which define a "LASTSTATE" definition for the home NIM, including the dimensions of the visual manifestation corresponding to the home NIM on line 38 (HEIGHT="134" X="616" Y="109") and indicates that the home NM is not collapsed upon startup (COLLAPSED="FALSE").

(99) The visual manifestation of home NIM 162 illustrates additional benefits and features of a home NIM in accordance with the present invention. When a user selects tab 176, a list of the NIMs that are present in the processed user profile associated with the user is displayed in viewer 180. As disclosed in more detail below, a user has the option to associate a collection of NIMs into an object termed a "pack". The pack references some subset of the NIMs associated with a user as well as associated state information. This arrangement includes, for example, whether a visual manifestation corresponding to each MM is displayed on output means 42 or not, the dimensions of each visual manifestation,

and the position of each visual manifestation. The name of each pack is stored in the processed user profile. A user reviews packs associated with the user by selecting tab 178 (FIG. 4). In FIG. 4, the user only has one pack, "Customized DotPack" 182. When the user selects pack 182, each NIM in the pack is restored in accordance with the state information stored in the pack definition.

(100) In total, FIG. 4 represents a typical visual experience provided by one embodiment of the present invention. In addition to home NIM 162, visual manifestations 184 and 186, corresponding to two additional NIMs in the processed user profile, are displayed. Visual manifestation 184 provides functionality that allows a user to manage an address book, schedule appointments, or create groups and plan activities. Visual manifestation 186 represents a NIM that provides time and date information.

(101) FIG. 5 shows the visual manifestation of home NIM 162 of FIG. 4 with tab 176 selected. Accordingly, each of the NIMs in the processed user profile associated with the user is listed in list 188. The user can activate any of the listed NIMs by clicking on the NIM name. In addition to the NIMs in list 188, home NIM 162 includes core NIMs that are defined in "session_config." In the "session_config" of Example 1, cores are found on lines 28 and 29. Specifically, line 28 provides the address of an XML-based definition for the add template functionality associated with button 172 in FIGS. 4 and 5, and line 29 provides the address of an XML-based definition for the help template functionality associated with button 174 in FIGS. 4 and 5.

(102) Importantly, the user can categorize NIMs using filter 190. Categories include such topics as sports, personal, weather, etc. Furthermore, the user can add NIMs to the processed user profile associated with the user as well as delete NIMs. There are a variety

of mechanisms that enable a user to add a NIM to the processed user profile. One mechanism is to receive links to NIMs from other users of system 10 (FIG. 1), as disclosed below. Another mechanism is to toggle button 172 in order to activate a visual manifestation associated with NIM 192 (FIG. 5).

(103) NIM 192 provides a system that enables users to add select NIMs to their user profile with a single click or keystroke sequence. NIM 192 includes tab 194 that allows the user to select premiere NIMs and a general tab 196 that allows the user to review a general catalog of NIMs that is present in NIM templates database 74 (FIG. 1). In one embodiment, when a user selects a NIM in list 198 (FIG. 5), the NIM is added to list 188 and is incorporated into the processed user profile associated with the user. In this way, the user can collect NIMs of interest to the user using a single mouse click. By illustration, consider the case in which a user selects the NIM "AnyDay Calender" in list 198. In response to this selection, a definition of the NIM "AnyDay Calender" is obtained from NIM templates database 74 and is copied directly into the processed user profile associated with the user. Furthermore, the title of the selected NIM, "AnyDay Calender" is added to list 188. Finally, a visual manifestation that corresponds to the NIM "AnyDay Calender" is displayed on output means 42 (FIG. 1). As a result, the display illustrated in FIG. 5 adopts the appearance illustrated in FIG. 6.

(104) In FIG. 6, the NIM "AnyDay Calendar" appears at the top of list 188. Furthermore, a control 101 associated with the NIM "AnyDay Calendar" in list 188 is filled, indicating that the NIM is currently active. Additionally, as illustrated in FIG. 6, a visual manifestation 103 corresponding to NIM "AnyDay Calendar" appears on the output means. The user has the ability to toggle this NIM between an inactive and active state by selecting control 101.

(105) In one embodiment, the user is provided with the option of (i) incorporating a NIM selected in list 198 into the processed user profile or (ii) transiently executing the NIM on client 20. Furthermore, when the user receives NIMs from other users, the user has the option to transiently operate the received NIMs on client 20. If the user decides to keep the transient NIMs at a later date, the user has the option to add the transient NIMs to the processed user profile at that time. Thus, in such embodiments, the user effectively has the option to "preview" NIMs before adding them to the processed user profile. This is advantageous because it reduces the chances of filling the user profile with undesirable NIMs. Such a feature is particularly advantageous in the case of novice or inexperienced users of system 10. Furthermore, one of skill in the art will appreciate that the concept of transient NIM execution raises the possibility of executing NIMs on a client 20 during a period of time in which the user is not logged into system 10. For example, consider a NIM that is executed on a client 20 after a user initiated response to a web page advertisement presented in web browser 36. Although the user is not logged into web server 58 and therefore does not have a processed user profile resident on client 20, the user can execute the NIM on client 20 on a transient basis. Furthermore, if the user wishes to add the transiently executed NIM to the user profile 76 associated with the user, the user can log into web server 58 and then add the NIM to the processed user profile that is delivered to client 20 as a function of the log in process.

(106) Another important feature of the present invention is the ability for users to share NIMs with each other. For example, if a user wishes to share the NIM "AnyDay Calendar" that was added to list 188 in FIG. 6, the user clicks "share" button 170 (FIG. 6). In response, panel 105 is displayed (FIG. 7). Because "share" button 170 is pressed while tab 176 is

active in the illustration provided by FIGS. 6 and 7, panel 105 lists each of the NIMs associated with the user. If, however, "share" button 170 is pressed while tab 178 is active rather than tab 176, panel 105 will list each of the packs associated with the user instead of each of the NIMs. Returning to the situation illustrated in FIG. 7, the user shares a NIM with other users by selecting the NIM to be shared from list 107 and then toggling button 109 "Share via email." In one embodiment, the user has the option to select multiple NIMs from list 107 using predefined keystroke operations. For example, in one embodiment, the user selects multiple NIMs by clicking on several of the NIMs in list 107 with a mouse button while depressing the "shift" button on the keyboard. When a user decides not to share a NIM and panel 105 is displayed, the user presses cancel button 111 and panel 105 is dismissed.

(107) When a user toggles "share via email" button 109 at a time when one or more NIMs in list 107 have been selected, the definition of each selected NIM is copied from the processed user profile associated with the user into a container and the container is sent to server 50 (FIG. 1). In the embodiment shown in FIG. 1, the container is received by web server 58. Web server 58 includes instructions for routing the container to sharelink database 78 where the container is stored. When the container is stored, a unique identifier is assigned to the container. Although a large number of different mechanisms for generating a unique identifier are practiced in accordance with this aspect of the invention, in one embodiment, the unique identifier assigned to the container upon storage in sharelink database 78 can be subsequently processed to form a URL address that specifically references the container within the context of system 10. In one embodiment, after a unique identifier has been assigned to the container, an e-mail program is launched

on client 20 and the user is requested to designate the recipients of the designated NIMs. Then, each recipient is provided with the unique identifier associated with the container in an e-mail message. When the recipient clicks on the unique identifier, a call is made for a copy of the associated container from sharelink database 78 and the container is delivered to the client 20 associated with the recipient.

(108) As is readily apparent upon review of FIG. 7, the user has the option to size and position the visual manifestation that corresponds to each NIM. Furthermore, by toggling controls, such as toggle button 101 (FIG. 7), the visual manifestation of a NIM is toggled between an on state and an off state. Such functionality is highly advantageous. First, by using this functionality, the user has the option to create unique arrangements. Second, NIM developers have the ability to control the default position and size of NIMs as well, and can therefore produce an arrangement of NIMs to further specialized purposes. Finally, because the NIMs of the present invention are not trapped in third party applications that have a set of undesirable features such banner ads, the utility and overall appearance of an arrangement of NIMs is enhanced and adopts an independent value. Using the technology disclosed in the present invention, the user collects an assortment of NIMs and arranges them in a customized fashion. The user has the option to "capture" favored arrangements into constructs known as packs, which have been briefly discussed previously.

(109) FIGS. 8 and 9A illustrate the formation of a pack using the arrangement of NIMs illustrated in FIG. 4. The process begins when the user toggles button 164 "Make Pack" in FIG. 4. In the embodiment illustrated by FIGS. 8 and 9A, panel 113 (FIG. 8) is displayed when the user toggles button 164 (FIG. 4). Panel 113 advises the user to open and arrange each of the NIMs that are to be included in a pack. In the case of FIG. 4, for example, such

an arrangement could include the arrangement of NIMs 184 and 186. The user indicates that specified NIMs are in a desired arrangement by selecting button 115 "Next" (FIG. 8). When button 115 is toggled, prompt 113 is terminated and prompt 117 is displayed to prompt the user for a name to associate with the designated pack. The user indicates that a name 119 has been provided for the pack by selecting "Done" button 121.

(110) In the embodiment shown in FIG. 8, the user further has the option to return to panel 113 and rearrange the specified NIMs before committing to pack creation by selecting the "Back" button 123. In the situation illustrated in FIG. 8, the user has provided the name "New DotPack." FIG. 9A illustrates the state of the visual manifestation corresponding to home NIM 162 after the user has selected "Done" button 121 (FIG. 8). Specifically, the name "New DotPack" is added to pack list 125 and tab 178 is activated to display the user pack list rather than the user NIM list that is displayed when tab 176 is activated. Furthermore, in response to the user selection of "Done" button 121 in FIG. 8, a reference to each NIM specified by the user is collected into a pack, along with some state information, and the pack is stored in the processed user profile associated with the user. Representative state information for each NIM stored in a pack includes whether the NIM was collapsed and the position of the NIM. In some embodiments, the state information includes the dimensions of the last visual manifestation corresponding to the NIM to have been displayed on output means 42.

(111) FIG. 9 illustrates pack 139, which is delineated with a dashed box. Pack 139 includes five NIMs 133. Each NIM 133 includes two primary components, a viewer 135 for viewing content and a frame 137 for providing user functionality. Each viewer 135 provides a platform for reviewing machine readable information, such as HTML, dHTML, images,

imbedded ActiveX and Java applications, JavaScript, CSS, Streaming Media, and/or Flash. Furthermore, in some embodiments, viewer 135 provides functionality for processing digitally recorded sound. Observe then that each NIM of the invention utilizes machine readable information that is easily retrieved from a specified address. If the content provider of this information desires to change the machine readable information, changes can be made and then delivered the next time that the machine readable information is addressed. This stands in contrast to prior art techniques in which updates to executable code can be relatively difficult to distribute.

(112) Each frame 137 provides functions and controls for manipulating the visual manifestation of the NIM 133 corresponding to the frame. For example, some frames 137 include a dismiss button 141. When the user clicks on dismiss button 141, the corresponding NIM 133 is dismissed. Additionally, some frames 137 include a menu button 143. When the user clicks on button 143, a menu appears. In one embodiment, the menu is a pull down menu and the menu items are defined in the corresponding NIM definition. In an abstract example, the NIM definition provides a menu definition of the format:

Menu 143-(I)-"Get more NIMs" URL

(113) where (I) refers to the N.sup.th element of the menu that is activated when menu button 143 is pressed, "Get more NIMs" is the menu element name, and URL is the event or web address that is called when the user selects the N.sup.th element of the menu associated with button 143.

(114) The developer has full control over all aspects of the appearance and functionality of NIM frame 137. Thus, a NIM developer has the ability to control, for example, the

functionality located at any corner of frame 137, functionality placed along the top or bottom of the frame, or on the sides of the frame. As an illustration, frame 137-3 includes a control button 145 that allows the user to resize the visual manifestation of NIM 133-3. One of skill in the art will appreciate that the bottom row of NIM frame 137 could have any number of buttons, including a forward control, a backward control, and similar controls, each of which provides the user with distinct functionality.

(115) An advantage of the present invention is that it is possible to embed commands that affect frames 137 in the content of the computer readable media delivered to frame viewer 135. The use of embedded commands provides NIM developers with powerful tools and additional flexibility. For example, a developer can use embedded commands, such as the menu command above, to design a NIM that has a context sensitive help menu. Each time a particular event occurs, the developer changes the content of the help menu using embedded commands. For example, when a sound file is delivered to a NIM, the sound file may be attached to a set of embedded commands that provide sound manipulation functionality in the form of a pull down menu. Elements of such a pull down menu include a command for saving the sound file to disk, commands for fast forward, stop, and play, and commands for sound enhancement. After the sound file has been played, embedded commands are used in this example by the developer to reset the menu associated with button 143 to some default state. One of skill in the art will appreciate the benefits and advantages of a frame 137 that is capable of being modified based on commands embedded in the machine readable media delivered to the corresponding viewer 135. The developer can use presentation tools, such as adding transient help buttons, resizing the visual

manifestation of the NIM, changing the frame border color, changing the title of the frame, and changing the frame border patterns, to create a more effective application.

(116) One of the advantages of the disclosed pack system is that it provides a convenient mechanism for rapidly assembling NIMs that track diverse sources of information. Furthermore, when a useful set of NIMs is collected into a pack, the user can share the pack with other users using the same procedure previously identified for sharing one or more NIMs. It will be appreciated that in some embodiments, NIMs are commercial applications and that appropriate use of packs provides an additional dimension for application development. Therefore, in some embodiments, pack recipients are subscribers to a service provided by a pack developer. In other embodiments, pack recipients are purchasers or licensees of packs. In still other embodiments, pack recipients receive packs developed by friends, family members, or business associates.

(117) The technology of the present invention further provides a set of NIM management tools to help a user manage displayed NIMs. In some embodiments of the present invention, these management tools are provided by visual management module 114 (FIG. 1). Two such NIM management tools are, in fact, properties that are associated with NIMs, namely magnetism and snapping. The property of magnetism is exhibited when a visual manifestation corresponding to a first NIM is dragged or moved near a visual manifestation corresponding to a second NIM. When this occurs, the two NIMs exhibit a magnetism that causes the first NIM to accelerate toward the second NIM. However, when the visual manifestations are within a predetermined distance of each other, the NIMs snap together. In one embodiment, the predetermined distance that triggers the two visual manifestations to snap together is a gap of about five pixels. While the above discussion describes the

principles of snapping and magnetism based on a pair of NIMs, there is in fact no limitation on the number of NIMs that can be snapped together and furthermore, the principle of magnetism is not dependent on whether a NIM is in fact snapped to another NIM or not.

(118) An additional management tool, illustrated in FIG. 9A, provides a mechanism for selecting multiple NIMs and for moving the NIMs in a coordinated fashion. In FIG. 9A, visual manifestations 162, 184 and 186 corresponding to respective NIMs are locked together. In response, halo 127 is drawn around the selected locked visual manifestations to graphically notify the user which NIMs are locked together. As mentioned previously, the user has the option to position NIMs as a coordinated group. For example, in one embodiment, when halo 127 is displayed and the user moves mouse arrow 129 after selecting one of the visual manifestations corresponding to a locked NIM, a target manifestation 131 is displayed to indicate to the user the position that the selected NIMs will be relocated to if the user clicks a mouse key. In some embodiments, target manifestation is a shadow image of the NIMs within halo 127 rather than the box depicted in FIG. 9.

(119) The present technology further provides additional methods for controlling visual manifestations of NIMs. For example, in one embodiment, the user has the option to select multiple NIMs by pressing a predefined key such as the keyboard "Ctrl" key, before selecting a specified NIM. While continuing to depress the control key, the user has the option to select additional NIMs and add the corresponding NIMs to a group. The user then has the option to move each of the NIMs in a single coordinated fashion as a group. Furthermore, by entering a designated keyboard or mouse sequence, the user has the option to move a single NIM even in situations where the NIM is in a group. In one embodiment in

accordance with this aspect of the invention, the user clicks the visual manifestation corresponding to a locked NIM that the user wishes to move in an independent manner and the user does not click the visual manifestation when the user wishes to move the NIM in a manner that is coordinated with the other NIMs. Additional features of the present invention include the option to select rows or columns of NIMs using specialized control sequences. For example, in one embodiment of the present invention, the user selects a column of NIMs by clicking on a NIM while holding down the alphanumeric character "c" on the keyboard.

(120) It will be appreciated that one advantage of the present invention is that it is possible to display multiple NIMs and that each NIM provides a specialized visual experience. Therefore, NIM alignment tools are advantageous because they allow users to quickly make room on output means 42 for additional NIMs and/or to produce highly styled NIM arrangements. Accordingly, the present technology provides a specialized feature to rapidly align NIMs. This technology is illustrated in FIG. 10. The technology works in conjunction with the tools for selecting multiple NIMs. In FIG. 10A, the user selects the visual manifestations 147, 149 and 151 corresponding to respective NIMs using, for example, the column select feature disclosed above. Then, the user pushes the selected NIMs against boundary 153. In one embodiment, boundary 153 is the horizontal or vertical edge of output means 42. In another embodiment, all visual manifestations corresponding to NIMs are displayed in a single viewport such as a window. In such embodiments, the horizontal and vertical edges of the window each represent a boundary. FIG. 10B illustrates how visual manifestations 147, 149, and 151 are automatically aligned when they are pushed against a boundary, such as boundary 153. In some embodiments, the relative alignment

between the visual manifestations is preserved even after the manifestations are moved in subsequent action by the user. The present technology further allows for the rearrangement of NIMs along a particular axis. For example, if NIMs are substantially oriented along a vertical axis as shown in FIG. 10B and the user wishes to realign the NIMs on the horizontal axis, all the user has to do is push the collection of NIMs against a horizontal border. For example, when the user pushes visual manifestations 147, 149 and 151 against border 155, the visual manifestations realign to conform to border 155 thus resulting in the view depicted in FIG. 10C.

(121) The features of the home NIM of the invention have been fully described. Attention presently turns to the architecture and operation of individual NIMs utilized in accordance with the invention.

(122) In one embodiment of the invention, after a user has logged into the system, as discussed above in connection with FIGS. 1 to 10, the user interface 40 displays the home NIM 162 as shown in FIG. 5. The home NIM typically includes a list of NIMs 188, referred to in FIG. 5 as "MyDots." These are NIMs which have been "collected" by the user. The list of collected NIMs, along with their associated definitions, is stored on the server in the user profile database 76, and downloaded from the application server 50 in response to a request from the client parser application 38. The local copy of the processed user profile is then further processed when the user collects or uses NIMs.

(123) Collected NIMs may be opened or closed by clicking on the control button next to the NIM's name or on the NIM's name itself, in list 188, and all NIMs may be closed by clicking on the "all off button` 166. The user may place NIMs into categories in a list of

categories 190, which can be edited by clicking on the-Edit" button 168. New NIMs may be added to the user's collection of NIMs by clicking on the "Get" button 172.

(124) FIG. 5 also shows a NIM 192 with a list of NIMs 198, which may be previewed and/or collected by the client 20. The user may preview or collect a NIM by clicking on the associated name of the NIM, e.g., "eHOW". The steps taken to provide the NIM to a user are shown in FIG. 11. After logging in (step 240) the user clicks on the name of a NIM, and the client parser application 38 sends a request including the NIM_ID of the NIM definition, to the applications server 50 via the transmission channel 44 (step 241). Alternatively, the user may click on a NIM link before logging in, for example if the link has been e-mailed to the user, and then, after clicking on the link, log in. In another aspect, the user could view, but not collect, a transient NIM without ever logging in.

(125) After the user is logged in and has clicked on the NIM, the applications server 50 retrieves the NIM definition from the NIM template database 74 using the NIM ID, in step 242, and provides it to the client 20 in step 243. The client 20 receives the NIM definition from the applications server 50 in step 244, and the client parser application 38 creates a frame in the display of the user interface 42 in step 245. In step 246, the client 20 requests the necessary content elements 94 stored at the URLs identified in the NIM definition from the corresponding content server 82. The content server 82 transmits the content 94 in step 247, and in step 248 the client parser application 38 places the content in the viewer, which is enclosed by the frame, allowing the user to preview the NIM. Alternatively, the client parser application 38 may simply collect the NIM, adding it to the user's processed user profile.

(126) The user may then view the NIM on the user interface display 42, and may interact with the NIM much in the same way as a user may interact with Internet content or web applications. This may change the NIM from its present, "raw" state to a used state reflecting alteration or use of the NIM by the user. For example, the user may direct the NIM to different content within the NIM if the NIM content enables the user to do so. Or, the user may provide information to the content server 82 which allows the NIM to be personalized. The user may additionally be given the option of changing the size of the frame.

(127) If the user collects the NIM, the NIM will be added to the user's list of collected NIMs such as the list 188 shown in FIG. 5. Additionally, the client parser application will add the NIM's definition to the processed user profile, and, on logout, send the processed user profile to the application server 50. Thus, the NIM's "state" will be preserved. Alternatively, the client parser application may collect the NIM automatically, without waiting for a user command, by adding the NIM definition directly to the processed user profile.

(128) If the NIM's state has been altered by the user or by the content--if for example, the user has directed the NIM to Internet content other than the initially-displayed content, provided personalizing information, or changed the properties of the frame, or if the content itself has caused an alteration in the NIM--this alteration will be reflected in the NIM definition stored in the user profile database 76. Information which personalizes the resulting content, instead of being stored in a "cookie" on the client's hard drive, can be stored as part of the NIM definition. This advantageously permits personalization of content, such as web

content that is associated with the NIM content and the user, without storing a cookie on the client 20.

(129) A user may also access a NIM which has been previously collected, and possibly altered by use as explained above. As previously described, the user profile 76 includes NIM definitions for NIMs which have been viewed and collected by each user. A screen shot showing NIMs 188, which have been previously collected by a user is shown in FIG. 5. The steps taken to provide the user with NIMs which have been previously collected are shown in FIG. 12.

(130) As discussed above, on login (step 250) the user's profile is retrieved by the client parser application 38 in the client 20 (step 251 and 252). The user's profile, stored in the user profile database 76, includes the NIM definition for each of the NIMs previously collected, and possibly altered, by each user. The NIM definitions, as discussed above, includes the NIM frame definition and the definition of the controls for filling the viewer within the frame with content. After log in, a local copy of the processed user profile is stored on the client 20, and this copy is further processed as the user collects new NIMs, or uses new or collected NIMs such that the NIMs are altered.

(131) When the user clicks on the name of a collected NIM (step 253) the client parser application 38 creates a frame in the display 42 of the user interface 40 in step 254. At step 255, the client 20 requests the necessary content elements 94 stored at the URLs identified in the NIM definition from the corresponding content servers 82, which provide the content 94 in step 256. It will be appreciated that these URLs need not be the same as the initialization URLs in the "raw" NIM definition stored in the NIM template database 74 on applications server 50, and in fact the content servers need not be the same content servers

corresponding to the initialization URLs. In step 257, the NIM parser application 38 places the content in the NIM frame, and the NIM is then fully opened.

(132) FIG. 13 illustrates a data structure for a NIM definition. As discussed above, a NIM is defined as a frame that contains a collection of controls, or functional units, such as a web rendering control or a GIF rendering control. The NIM frame surrounds a viewer, which displays the addressed content. The MM has a defined layout or arrangement of the controls, and defined initialization input data, e.g. data and URLS, for each control or element, in the NIM. NIM definitions are available to the client parser application via NIM links. The NIM links "point" to NIM definitions, which include all the information needed to build a NIM frame and fill the NIM with NIM content. Thus, NIMs links are easily collected, associated into packs, and shared by users.

(133) In one embodiment, the NIM definitions are defined using Extensible Markup Language (XML), so that the NIM as a whole--the frame and the content within the viewer--is advantageously as flexible as standard web content. NIMs are extremely flexible, because the definition of the NIM is content, rather than compiled code. The NIM definition defines the structure of the NIM, and everything that is visible in a NIM is based on standard Internet content, such as HTML, dHTML, or GIFs, and is referenced or pointed to by the NIM definition. An "application"-type NIM, such as a web calendar or web mail, may be changed by the user, by the content provider, or by other content, while advantageously avoiding the need to distribute and support a hard-coded compiled application. The definition of a NIM thus includes everything that is needed for the NIM to be rendered and filled with Internet content.

(134) As shown in the exemplary embodiment of FIG. 13, the definition of a NIM includes tags that identify the NIM 270, define and configure the NIM frame 271, specify and layout the controls 273 in the NIM viewer, and specify parameters to initialize all the NIM's components with content or data.

(135) In one embodiment, a NIM is identified by three ID strings 270: GlobalID, Domain and Kind. A GlobaliD is used when the MM definition is within a share. It is unique with respect to other NIM tags in the share. A NIM's domain is a unique label for the owning company or developer of the NIM, such as "dodots.com ." Finally, a NIM's kind, which is specified by the NIM's developer, is a helpful identifier for finding the NIM, but need not be unique. Examples of possible NIM kinds include "mp3", "scriblets," and "calculator." As discussed above, a NIM definition will typically be written in a format which facilitates sharing of data over the Internet, such as XML. An XML specification for the NIM identification strings, for one embodiment of the invention follows. The bold text identifies NIM definition data, while the remaining text describes the data.

GLOBALID="string" Used only within <SHARE> tags. This GLOBALID must be unique with respect to other <NIM> tags in this <SHARE>.

DOMAIN="string"

(136) Unique label for the owning company of this NIM. In theory, NIMs may be limited to communicating with NIMs only from their own domain.

KIND="string"

(137) Helpful identifier for finding such a NIM from another NIM. Does not have to be unique.

(138) The NIM definition also includes the definition of a frame 271, which specifies the frame size and shape, and optionally the frame orientation and/or location on the user's screen. The space within the frame is the control space or viewer; visible controls are distributed within the control space or viewer.

(139) The NIM definition may optionally include controls for: a titlebar; a NIM menu with flexible menu entries; an exit button; and a bottombar. A typical layout for these components is: titlebar at the top of the control space, with menu on the left and exit button on the right, and the bottombar at the bottom.

(140) The titlebar component gives the user a place to grab and drag the NIM in a windowed environment. In one embodiment, it is implemented as a GIF rendering control that can be targeted to a local or remote titlebar image. The titlebar will preferably have a fixed height and width that is a function of the NIM's width. The titlebar is preferably capable of being located at any position on the periphery of the NIM. Overlay text can also be specified to layer on top of the titlebar image. The bottombar may be implemented in a similar fashion, but typically will not include text overlay. The titlebar and bottombar may be filled in with initialization data from a fixed data file, or alternatively with Internet content from, example, an initialization URL.

(141) In one embodiment, a menu definition 271 is also included in the NIM definition. The menu includes items and actions of the NIM provider's choosing. For example, menu items may include the title "browse" associated with the action of targeting a full-screen browser or another NIM, and retrieving content for that browser or NIM from a specified address

such as a URL. Logging off, or directing the NIM to another address or URL, are also possible menu action items. Menu action items that require communication of messages between the NIM and another NIM may also be provided--for example, opening another NIM, or changing the content of another NIM that is already open. Communication of messages between different parts of the system is discussed below.

(142) An XML specification for a frame, titlebar, bottombar and menu, for one embodiment of the invention follows:

<FRAME>

<TITLE>

TEXT="string"

JUSTIFY=-"LEFT"|"CENTER"|"RIGHT"

COLOR="#XXXXXX"

PIXELWIDTH="integer"

Width in pixel units. Overrides WIDTH attribute.

PIXELHEIGHT="integer"

Height in pixel units. Overrides HEIGHT attribute.

WIDTH="integer"

Width in NIM units. Default value is 1.

HEIGHT="integer"

Height in NIM units. Default value is 1.

X="integer"|"LEFT"|"CENTER"|"RIGHT"

Initial X position in screen coordinates. Default is center.

Y="integer"|"TOP"|"CENTER"|"BOTTOM"

Initial Y position in screen coordinates. Default is center.

FIXWIDTH="TRUE"|"FALSE"

Default is false.

FIXHEIGHT="TRUE"1"FALSE"

Default is false.

BACKGROUND COLOR="#XXXXXX"

Default is white.

<TITLEBARIMAGE>

JUSTIFY="LEFT"|"CENTER"|"RIGHT"

TILELEFT="integer"

TILERIGHT="integer"

NORMAL="URL"

DOWN="URL"

HOVER="URL"

INACTIVE="URL"

<BOTTOMBARIMAGE>

JUSTIFY="LEFT"|"CENTER"|"RIGHT"

TILELEFT="integer"

TILERIGHT="integer"

NORMAL="URL"

DOWN="URL"

HOVER="URL"

INACTIVE="URL"

<MENU>

Contains zero or more <ITEM> tags.

<ITEM>

TITLE="string"

TOOLTIP="string"

ICON="URL"

ID="string"

Must be unique.

<ACTION> RECIPIENT="address" MESSAGE="string"

(143) As shown in FIG. 13, the NIM definition also includes layout and definition of the controls 273. A control may be visible and render some sort of visual or text display, either static or dynamic. A control may be hidden, for example a functional element that is not necessarily visual such as a Java control. The control definition 273 includes identification of the types of controls, the layout of the controls, and initialization information. In one embodiment, NIM controls are specified and identified by class, kind and ID. Class defines the type of NIM control and is not unique. Kind is a useful identifier selected by the developer, and again is not unique. The NIM ID is unique within a user's processed profile.

(144) Different classes of controls may be used. For example, a control may be a web rendering object, which can render web content such as HTML, dHTML, images, imbedded ActiveX and Java applications, JavaScript, CSS, Streaming Media, and/or Flash. Alternatively, a control may be any object capable of rendering any kind of computer

readable media, such as a GIF rendering object or an custom-designed object to display a particular kind of information. Alternatively, a control may be an object capable of processing any kind of application logic, such as a Java module. For example, an on-line brokerage firm could implement a custom stock-chart-rendering control, and define a NIM to use this control.

(145) As discussed above, the control space is located within the frame, and one or more controls can be flexibly positioned within the control space, and these controls may include a titlebar and a bottombar, as well as other controls. The location of controls is specified by the layout in the definition of the controls 273 within the NIM definition. In one embodiment, the controls are laid out or positioned within the NIM frame according to a flexible grid. In this embodiment, the NIM definition allows the control space to be subdivided into equal vertical and horizontal units, and then for the controls to be positioned and sized within the control space.

(146) A control definition will typically include initialization data. For example, where a control is a web rendering object, the definition will include initial URLs. When the NIM is opened, the control will navigate to the initial URLs to obtain content and render the NIM. If the control is a GIF, the control could retrieve the GIF file from a content server or from the application server. A NIM definition may optionally include additional tags identifying initialization parameters for different platforms: for example, a URL is suitable for a PC, but a "P-URL" may be provided as well, pointing to content suitable for users viewing NIMs through a personal digital assistant (PDA) or similar device.

(147) Controls are typically installed on the applications server, and may be updated after installation by the applications server. The home NIM code, downloaded when the client

becomes "NIM-enabled," includes the then-existing controls. Controls are updated as new controls are installed on the server or when a user requests a NIM that requires a new control. The server may then may download such updates to the client parser application, for example on log-in. The NIM framework allows any control to be positioned and initialized in a control space in a NIM, as discussed above.

(148) An XML specification for control definition and layout, in accordance with one embodiment of the invention, follows:

<CONTROL_LAYOUT> Contains zero or more <CONTROL> tags.

WIDTH="integer" Divides control space into this many evenly spaced columns. Default is 1.

HEIGHT="integer" Divides control space into this many evenly spaced rows. Default is 1.

WIDTHSCALES="TRUE"|"FALSE" Default is true.

HEIGHTSCALES="TRUE"|"FALSE" Divides control space into this many evenly spaced columns. Default is 1.

<CONTROL>

CLASS="string" Class may be "Browser," "GIF reader," or other object for rendering

computer readable media.

KIND="string"

ID="string" Must be unique with other controls in this NIM.

LEFT="integer" X position of the control in container unites. Default is 0.

TOP="integer" Y position of the control in container units. Default is 0.

WIDTH="integer" Width in container units. Default is 1.

HEIGHT="integer" Height in container units. Default is 1.

URL="URL" This is read if and only if this control is of class "Browser". This is the URL to which this control navigates. Otherwise, may include address for other control content, e.g. GIF address in applications server.

<CATEGORIES> Contains zero or more <CATEGORY> tags.

<CATEGORY> Adds the NIM to this category. This is the only way categories are specified; i.e. there is no master category list.

NAME="string" This is the name of the category.

(149) A NIM definition may also optionally include home NIM categories 274. A home NIM category used by home NIM 204 is a convenient way for a user to keep track of collected NIMS. When a user adds a NIM to a category 204, the category is added, as a string element, to the categories element 274 of the NIM definition in the user profile. For example, a user may categorize a particular NIM as "entertainment," or "news," or "reference." This category will then be added to the categories element 274 of the MM definition.

(150) A NIM definition may also optionally include an events element 275, which defines actions to certain NIM events. For example, the OnClose event, when a NIM is closed, may be assigned a specific and targeted action, similar to a menu item. An XML specification for the event element in a NIM definition, in accordance with one embodiment of the invention, follows:

<EVENTS>

<ONCLOSE>

(151) Executes this action list on close.

<ACTION_LIST>

(152) Contains zero or more <ACTION> tags.

<ACTION>

RECIPIENT="address"

MESSAGE="string"

(153) A sample NIM definition, in an XML file format in accordance with the above specification, follows:

<NIM DOMAIN="calculator" KIND="basic">

<FRAME CLASS="Standard" BACKGROUNDCOLOR="#FFFF00"

WIDTH="6" HEIGHT="4" FIXWIDTH="TRUE" FIXHEIGHT="TRUE">

<TITLE TEXT="Basic Calculator" COLOR="#000OFF" JUSTIFY="LEFT"/>

<TITLEBARIMAGE JUSTIFY="LEFT" TILELEFT="1" TILERIGHT="1"

NORMAL=" " INACTIVE=" " HOVER=" " DOWN=" "/>

<BOTTOMBARIMAGE JUSTIFY="LEFT" TILELEFT="1"

TILERIGHT="1" NORMAL=" " INACTIVE=" " HOVER=" " DOWN=" "/>

</FRAME>

<MENU/>

<CONTROL LAYOUT WIDTH="1" HEIGHT="1">

<CONTROL CLASS="Browser" KIND="A" ID="1" TOP="0" LEFT="O" WIDTH="1" HEIGHT="1"

URL="http://www.dodots.com /dots/Calc/CALCULATOR2.htm"/>

</CONTROL LAYOUT>

</NIM>

(154) The first line of this definition establishes the identification of the NIM definition, as discussed above: it is in the domain "calculator," and the kind of display is "basic."

(155) In one embodiment, the domain will be the domain name associated with the content provider. The domain name is a unique label for the provider or developer of the NIM. The NIM's "kind" is a helpful identifier for locating the NIM, and need not be unique. A NIM may also be identified using a GlobalID, when the NIM is being shared. Since the NIM defined by this XML file is not being shared, it does not have a GlobalID.

(156) The second line of the example XML NIM definition establishes the size and appearance of the NIM frame, defining a NIM viewer in which the NIM content will be placed. The third line ensures that the height and width of the frame are fixed--that is, the size of this frame cannot be adjusted by the user. The fourth and fifth lines establish the title of the NIM--"Basic calculator"--and its location. The next four lines establish the location and placing of the titlebar and bottombar, and relevant images, e.g. mouse-over. Thus, the first part of the example NIM definition defines the NIM frame. The definition of a frame, titlebar, menu and other aspects distinguish a NIM from a browser--the content provider has control over the frame size and every aspect of the NIM's appearance, whereas when a browser is used, the content provider has to adapt to the browser display size, and browser titlebar, menu, logo and other aspects cannot be controlled by the content provider.

(157) The rest of the NIM definition identifies, positions, and initializes the NIM's controls, which, in this case, are contained by the NIM frame. In this example, the next few lines establish that a single control will start in the upper left corner of the NIM viewer, that the control is of the type "browser," or web-rendering, and that the initialization URL for the control is www.dodots.com /dots/Calc/CALCULATOR2.htm. This URL is typically referred to as the "initialization URL," because it is where the NIM looks for NIM content when it is opened. Where the control is of type "browser," the content will typically be HTML content. However, any standard Internet content--HTML, dHTML, flash, streaming media, or Java, for example--may be used. As discussed earlier, a control, could include types other than a browser. The final two lines of the XML file close the definition file.

(158) It will be appreciated that the NIM is designed such that content consumes the entire frame. In one embodiment, the content for the corners of the frame--the menu and the exit

button--and the frame sizing images are served by an application server and referenced when the user logs in. Everything else is developed and served by a separate NIM developer. This differs fundamentally from the current approach to providing web content, in which there is a strong distinction between the viewer application--the browser--and the web page or web content. Using present browser-type technology, the content is trapped within the viewer. To obtain a cohesive application feel and access to application features, the current alternative is to develop custom client applications. NIMs allow a developer to provide an application feel without developing custom client applications.

(159) NIMs and the client parser application have a messaging architecture--the NIM messaging architecture, or NMA--that enables NIMs, controls, and the client parser application to communicate. Messaging, in combination with the NIM definition, gives the content of a NIM access to the application/rendering program--the client parser application--and to other NIMs, allowing true application behavior. The NIM definition, discussed above, is accessible, flexible, and may be changed by a NIM or a user while the NIM is in use, even after it's been rendered.

(160) The content provider, the user, or other NIMs can change a NIM. For example, the content--which includes the titlebar and menu elements--may be changed by the NIM provider by simply enhanced NIM content, using 20 messaging, so that NIMs can exhibit true application behavior. For example, an online brokerage firm can go beyond providing a NIM that renders stock tracking charts, and allow users to trade on-line via a second NIM that can interact with other NIMs such as the first NIM to help facilitate the activity.

(161) A NIM can be changed by its content, or by another NIM, using messaging. This enables a NIM to, for example, notify a user of events, such as a change in content. A NIM

could, for example, remain open, but in a collapsed mode, until a particular event occurs, and could then either expand to normal size, or open another related NIM. For example, if a particular stock hits a predetermined price, the stock tracking chart NIM discussed earlier can notify the user by expanding, by popping up a message, or by opening another NIM (such as a stock trading NIM). Alternatively, the NIM could notify the user of a particular event by coming into focus or changing size or content. These changes could also be made by sending messages from a content or applications server to the NIM.

(162) As illustrated in FIG. 14, all elements of the system can send and receive NMA messages. Message routing between NIMs, from a NIM to itself (that is, between e.g. the frame elements and a control, or one control and another), and from NIMs to the system, are handled by a message interface module 106, which is part of the client parser application 38 in the client 20. In one embodiment, the message interface module 106 resolves addressing queries, executes system-level commands from the NIMs such as "close all NIMs," and passes messages between NIMs. The message interface 106 may also communicate messages to controls such as the browser class controls, for example "navigate the addressed NIM's browser control to the argument URL." Alternatively, the message interface module 106 may route a message to an application or content server (not shown in FIG. 14) for expanded functionality. In one embodiment, the message interface module 106 uses an HTTP request to access an application program interface (API) call, with data optionally being sent and received in XML format. For example, the message interface 106 could send a message providing user input, such as credit card information to a credit processing module on a web server 82.

(163) In one embodiment, an NMA message has two components: a recipient, or address, and the message body. Both are represented as strings. The address may specify an exact NIM, a kind of NIM or control, a domain, or the system, meaning the overall home NIM display. For example, in one embodiment, the address may be in the form:

#<NTM specifier>:<control specifier>

(164) if the communication is between NIMs in the same domain, or

#<domain>:<NIM specifier>:<control specifier>

(165) or

#<domain>:<NIM specifier>:<NIM id>:<control kind>:<control id>

(166) if the communication is between NIMs in different domains, or

#system

(167) if the communication is to the system.

(168) In one embodiment, if the address is not properly specified it defaults to #system. The message interface 106 in the client 20 can flexibly allow, restrict MM addressing or sending messages--for example, the message interface can ensure that only certain NIMs can send message to a particular NIM. This allows NIM developers to develop coordinated NIMs that can interact, by sending messages to e.g. change content or open one another, without allowing other NIMs to interact with their coordinated NIMs.

(169) In one embodiment, a specifier in n address may be the unique identification of the NIM, control or domain in question: <specifier>:=<ID>. Alternatively, where the kind of NIM or control receiving the message is important but the specific NIM or control is not, the specifier may address a message to the closest matching recipient using a search criteria: <specifier>:=<kind>#<search criteria>. The kind should be a type of NIM or control that is installed in the system. Search criteria may be, for example, "any," "open," or "closed." Finally, where the control is being specified, the specifier may be a symbol, such as ".", indicating that the message is addressed to the sending NIM. For example, in one embodiment, the address #system sends the message to the system. The address #. sends the message to the NIM which sent the message. The address #7 sends the a message to the NIM with the identifier 7. The address #B#open:5 sends a message to the control with the identifier 5, in the first open dot of kind B found in the system.

(170) In one embodiment, certain NIMs will have privileges to send particular messages to certain NIMs, and not to other NIMs. For example, NIMs in a domain may be permitted to send control messages to other NIMs in the same domain, but not to NIMs in other domains. Thus, a NIM provider may have control over messaging between NIMs in his domain, and prevent NIMs in other domains from changing NIMs in his domain. Alternatively, NIM providers may coordinate with NIM providers in other domains, permitting certain messaging privileges between some of the NIMs in their respective domains. The HomeNIM and other system NIMs, such as the login NIM, which are in the system domain, may have certain messaging privileges that no other NIMs have, such as logging the user out or closing all the currently-open NIMs.

(171) In one embodiment of the invention, the second part of the message, the body, is represented as a string of characters. Messages may be sent to the client parser application, to the frame of a NIM, or to a control. Messages may be specific, defined messages, as shown in the following examples, or may be any javascript, which may be sent in or out of NIM content. Examples of defined messages to the client parser application, in one embodiment, are:

Application Message<arg>	Function
Refresh	Refreshes the user's profile.
#have-NIM <NIM-address>	Check if the user has the specified NIM as part of the user's profile.
#delete-NIM <NIM-address>	Remove the specified NIM from the user's profile.
#get-screen-width	Returns the width of the screen.
#get-screen-height	Returns the height of the screen.
#close-all-NIMs	Closes all open NIMs.
#get-NIM-ids <NIM-address>	Returns the NIM ID of the specified NIM.

(172) Messages can also be sent from a NIM to itself, or to another NIM, and the identified actions or functions, specified in the body, are performed on the receiving NIM. The following are examples of messages to a NIM for one embodiment of the invention:

Defined NIM Message<arg>	Function
#set-size<width><height>	Sets the size of the NIM.
#set-width<width>	Sets the width of the NIM.
#set-height<height>	Sets the height of the NIM.
#set-position<x-pos><y-pos>	Sets the position, of screen, of the NIM.
#set-title<title>	Sets the title of the NIM.
#collapse	Collapses, but does not close, the NIM.
#uncollapse	Uncollapses the NIM.
#set-user-sizable<width true/ false><height true/false> sizeable	Establishes whether the NIM is by the user
#set-background-color<color>	Sets background color of the NIM.

#set-title-text-justify<justify-keyword>	

(173) The above examples of NIM messages may, in one embodiment of the invention, be sent to a NIM by another NIM. For example, a user may provide input to a NIM, for example a stock tracking chart NIM, indicating an interest in another NIM, such as a stock trading NIM. The current NIM may then send an "open" message to the second NIM to open it, if it wasn't already open. The current NEM may then send a navigate message (see below) that may include an argument, such as a URL or other content pointer, so that the second NIM could be opened to a specific URL indicated by the first NIM.

(174) Finally, messages may be sent to a control of a NIM, either by the NIM itself, another control, the HomeNim system, or another NIM. As examples, the following control messages are provided in one embodiment of the invention:

Defined Control Message	Function
##<any javascript>	Run any javascript in the control, e.g. javascriptin a browser-type control.
#show	Set visibility control within a NIM.
#hide	Set invisibility of a control within a NIM.
#get-size	Get size of control.
#get-address	Query for unique ID of a control by kind.
#navigate	Navigate the control.

(175) Control messages may be used by a NIM, addressed to its own control, or to the control of another NIM. Control messages may also be sent by the HomeNIM, or by the applications server or content server.

(176) An example of a message to a NIM is window.external.PostMessage("#.:","#collapse)" this is a message from a NIM, to itself, collapsing the NIM. Another example of a message

from a NIM to another NIM is "window.external.PostMessage("#mp3#any:","#open")", which is addressed to a NIM with the kind "mp3," but only if it is in the same domain as the sending NIM. The body of the message instructs the message of type mp3 to open. An example of a javascript message is "window.external.PostMessage("#mp3#any:1", "33play( )")", which sends a message to the mp3 NIM control that calls the javascript function play( ).

(177) The operation of the home NIM and individual NIMs has been described. Attention presently turns to different techniques used in accordance with the invention 25 to host NIMs.

(178) As shown and described in relation to FIG. 1, the application server 50 includes a NIM Management Modulel 12, a NIM Templates Database 74, a ShareLink Database 78, and a user profile database 76.

(179) FIG. 15 is a diagrammatic illustration of an embodiment of the NIM Management Modulel 12. NIM Management Modulel 12 may contain the NIM Templates Database 74 and the ShareLink Database 78, discussed in further detail below. NIM Management Module 112 may also contain content 402 for filling in a NIM frame or for rendering Internet pages. Alternatively, content 402 may be stored elsewhere, such as on a Web server similar to the server 82 shown in FIG. 1. As discussed above, the content preferably contains Internet content such as HTML (Hypertext Markup Language), dHTML, and images.

(180) In addition, Management Modulel 12 preferably contains executable procedures 403 for controlling and managing the NIM system. These procedures 403 may include: a Collection procedure 404 for obtaining new NIMs; a Sharing procedure 405 for sharing

NIMs with others; Development procedures 406, such as a DevZone procedure 407 and a DevNIM procedure 408, for creating, modifying or deleting NIMs; Publishing procedures 409, such as a PubZone procedure 410 and a PubNIM procedure 411, for publishing NIMs so that they are publically accessible; and Administration procedures 412, such as an AdminZone procedure 413 and an AdminNIM procedure 414, for administering the system. It should be noted that the NIM sharing procedure, discussed in further detail below, may be processed by either the Server module (102 of FIG. 1), or the Sharing procedure (405 of FIG. 15). Control and management of the server and the NIM Management Modulel 12 components will now be discussed in further detail.

(181) FIG. 16 is a diagrammatic illustration of an embodiment of the NIM Templates database 74. In this embodiment, NIM Templates database 74 primarily stores XML NIM definitions in their initial unmodified state as initially designed by a partner and which have not been altered by a user in any way. The unmodified NIMs are also referred to as "Raw NIMs". The NIM Templatesdatabase 74 is used as the starting point for the development of NIMs which may later be customized or modified by a user, developer, or system administrator, as discussed below.

(182) For ease of explanation, the individual or organization that controls the server (50 of FIG. 1) will hereinafter be referred to as the system provider and the individual or organization who supplies the content will hereinafter be referred to as a partner. It should be understood that a provider, partner, user, developer, and administrator of the system may be distinct entities, the same entity, or a combination of both. Furthermore, as discussed above, each of the above entities is assigned access rights or privileges which permit or forbid that entity from performing different actions on the system.

(183) FIG. 16 illustrates an embodiment of a NIM Template database 74. A NIM identification number (NIM_ID) 416 is stored in a NIM Template table 41.5 in the NIM Templates database 74. A Raw NW, identified by its NIM_ID, includes a plethora of RAW NIM characteristics, including, but not limited to, a Raw NIM creation date 417, which indicates when the NIM was created and is useful when searching for NIMs created during a specific time, a Raw N.I.M definition module 418, and the NIMIndex categories 422 in which the NIM has been categorized. Each NIM when created in typically classified into one or more NIM Index categories by the developer, such as "Applications", "Business", "Entertainment" and "News".

(184) Each NIM is fully configurable and definable. The NIM definition module 418 contains details defining the NIM, such as the look-and-feel 419, of the Raw NIM, initialization URLs (Uniform Resource Locators) 420, and a location 421 of where the developer would like the NIM to open on a users's computer screen. The look and feel of the NIM is the appearance and function of the NIM interface. The look and feel may characterize the frame or skeleton layout, the graphics used to represent certain functions, such as opening and closing the NIM, whether the frame is sizable, and the appearance and operation of menus in the frame.

(185) The definition module 418 may also contain Initialization URLs 420 which reference resources containing content. The content resources may be HTML (Hypertext Markup Language), dHTML, images, programs such as Java applets, or any other file supported by HTTP. The Initialization URLs 420 contains the name of the protocol required to access the resource, a domain name that identifies a specific computer on a network, such as the Internet, and a hierarchical description of a file location on that specific computer. These

files or resources are then used by the home NIM to fill in the frame and controls with content. In addition, the definition module 418 may contain other details such as the location 421 on a user's computer screen where the NIM should initially open.

(186) The NIMIndex may be used to search for, learn about, and collect NIMs. The NIMIndex is typically accessed from either a Web browser, such as Internet Explorer.RTM. or Netscape Navigator.RTM. or from the home NIM, 108 of FIG. 1. A user may search for NIMs by, or according to, any field of the NIM Templates table 415 via the NIMIndex.

(187) A user accessing the NIMIndex from a Web Brow typically navigates to a main NIMIndex web page such as that shown in FIG. 17. FIG. 17 is an illustration of the main NIMIndex Web page 423. A user typically navigates to main NIMIndex Web page 423 from a NIM home-page (not shown), or while anywhere within the NIM Web site by clicking on the "Collect the DOtS.TM." link 424 in a menu 425. A user may search the NIMIndex by entering a search term in a form 426 and clicking on the "Search" button 428 which implements the Collection Procedure (404 of FIG. 15) to search the NIM Templates database (74 of FIG. 15) for NIMs that match the query. In one embodiment, the NLMIndex may be searched by NIM title, NIM description, or partner, as shown in the pull down menu 430. The user may also browse the NIMIndex by clicking on a link 432 to a NIMIndex category 434 which will navigate the user to a NIMIndex category Web page 440, as shown in FIG. 18.

(188) FIG. 18 is an illustration of a single NIMIndex category, the "Applications" category 435, shown in FIG. 17. A list of sub-categories (not shown), as well as a list of NIMs 442 and their short descriptions are shown in FIG. 18. A user may click on the "more detail"

link 444 to be taken to a page displaying a full description (discussed later in relation to FIG. 29B) of the NIM, shown in FIG. 19.

(189) Navigation of the Internet generally occurs through the use of URLs (Uniform Resource Locators), which are the addresses of files or resources accessible on the Internet. The type of resource depends on the Internet application protocol. Using the World Wide Web's protocol, the Hypertext Transfer Protocol (HTTP), the resource can be an HTML (Hypertext Markup Language) page, an image file, a program such as a Java applet, or any other file supported by HTTP. The URL contains the name of the protocol required to access the resource, a domain name that identifies a specific computer on the Internet, and a hierarchical description of a file location on the computer and usually takes the form: "URL=protocol://machine.name[:port]/directory/document.name`?[&argu- ments] " The "protocol" is the Internet protocol used to reach the document or resource. On the Web, the "protocol" is typically HTTP, but it can take any number of forms, such as ftp (file transfer protocol), file (a local file), gopher (gopher protocol), mailto (electronic mail address), news (Usenet news), telnet and tn3270 (interactive sessions), wais (wide area information servers), or the like.

(190) The "machine.name" is the name of the host come where the document resides (such as www.NIM.com). The ":port" portion of the address is optional and is only necessary when a resource is accessible through a non-standard TCP port number. Although the standard port number for HTTP is 80, there are numerous Web servers on the Internet that use non-standard ports, such as port 8000.

(191) The NIM system, however, may also utilize a proprietary NIM protocol. An example of a URL using the proprietary NIM protocol is: "NIMS:?NIMTemplate=<N/M_ID>"

(192) The NIM protocol URL is used to collect, distribute, and share NIMs. When collecting NIMs the NIM protocol URL is referred to as a NIMLink. When distributing or sharing NIMs the NIM protocol URL is referred to as a ShareLink.

(193) The "NIMs:" term defines the NIM protocol or scheme and is always followed by a colon. The "?NIMTemplate=<NIM_ID" is an argument, where a dollar sign (5) and a question mark (?) are used to denote path and/or search elements. It should be noted that no path is supplied (i.e.: "//path/to/something"). The argument instructs the client parser application (38 of FIG. 1) how to handle a user's selection of a NIM protocol URL and what the NIM protocol URL must do. For example, to obtain a NIM, the argument might read "NIMTemplate=123`, to obtain a Share (discussed below) the argument might read "Share=123", to obtain a Pack (discussed below) the argument might read "Pack=123", etc. The argument can be used to cause the client parser application to do anything within it's system of functionality by specifying new argument sets to build new types of special client parser application links.

(194) In one embodiment, the address for where the client parser application (38 of FIG. 1) searches the system (10 of FIG. 1) for the NIM Template or ShareLink database (74 and 78 of FIG. 1) is specified within the processed login script or session_config, although it could alternatively be specified within the NIM protocol URL. When a user clicks on a NIM protocol URL (from any where you can place and click on a link, for example in a browser, in a NIM, in email, in a document, etc.), the client parser application processes the NIM protocol URL in the same manner as a browser processes HTTP links and an email program processes mailto links.

(195) When a protocol URL takes the form of a NIMLink, the client parser application responds by obtaining the NIM definition from the NIM Template database, optionally adds the NIM to the user's processed user profile (unless the NIM has been specified to be opened in transient mode, which may be specified in the argument), and optionally opens the NIM on the user's display screen.

(196) In one embodiment, by default, unless specified otherwise, a NIM will be added to a users collection (transient--false) and will be opened (open--true). A NIMLink with arguments may look as follows: "NIMS:?NIMTemplate&transient=true" or "NIMS:?NIMTemplate&open=false". More than one additional argument could be added by appending another argument to the URL which may read as "&argument=value".

(197) A ShareLink (discussed below) is similar to a NIMLink and may read "NIMS:?share=123", where 123 is the SHARE ID referencing the share module within the ShareLink Database. Pack Links (discussed below) typically read as "NIMS:?pack-123", where 123 is the PACK_ID referencing a pack module within the NIM Template Database.

(198) FIG. 19 is an illustration of a full description of NIM content 446. A graphic of the opened NIM may also be displayed 448.

(199) Once the user decides that he would like to add a NIM to his home NIM, the user clicks on the "get it now" or "Get This Dots.TM. NIMLink 450 (FIGS. 18 and 19) which either runs the Collection procedure (404 of FIG. 15) which obtains that NIM's NIM definition module (418 of FIG. 16) from the NIM Template table (415 of FIG. 16), or opens another Web page as shown in FIG. 20.

(200) FIG. 20 is an illustration of a Web page 452 which might be displayed to the user once the user has clicked on the NIMLink 450. The user is presented with an option of either

collecting the NIM 456, or if the user does not have the home NIM application, the user may first download the home NIM by clicking on "Get the homeDotIm" 454. Once the user clicks on the download the NIM button 456, the Collection procedure (404 of FIG. 15) obtains that NIM's NIM definition module (418 of FIG. 16) from the NIM Template table (415 of FIG. 16).

(201) The Collection procedure (404 of FIG. 15) transmits the NIM definition to the user's home NIM, which optionally opens the MM and saves the NIM definition module (418 of FIG. 16) on the user's local processed user profile. All NIM definition modules (418 of FIG. 16) on the user's computer may subsequently be saved to the user profile database, as discussed earlier in this writing. Alternatively, a "preview" button may be provided which transiently displays the NIM on the user's computer screen without adding the NIM to the user's local processed user profile. The user may also search the NIMIndex from their home NIM.

(202) FIG. 21 is an illustration of the main home NIM graphical user interface (GUI) 464, similar to that shown in FIG. 5. The home NIM displays a list of all NIMs 466 that the user has collected. Furthermore, any NIMs that the user has collected in groups or packs, can be accessed by clicking on the "My Dotpacks'" tab 468. One way to obtain new NIMs is to click on the "Get" button 470, which opens the NIM shown in FIG. 22.

(203) FIG. 22 is an illustration of a get new NIM GUI 474. A list of all NIMs 476 (or a featured subset) that may be collected by the user are displayed. Clicking on the "More Dots'" tab 478 (shown in FIG. 23) displays further NIMs which may be collected.

(204) When a user selects or clicks on any of the NIMLinks 480, NIMLink 480 references the NIM_ID (416 of FIG. 16) for that NIM in the NIM Templates database (74 of FIG. 16). The

collection procedure (404 of FIG. 15) receives the NIM_ID (416 of FIG. 16) from the user, locates the NIM definition module (418 of FIG. 16) corresponding to that NIM_ID in the NIM templates database, and transmits the NIM definition module to the user's computer. That NIM may automatically be opened on the user's computer screen. The NIM is saved to the user's list of NIMs on their home NIM (466 of FIG. 21), and the NIM definition module is saved in the user's local processed user profile. Alternatively a "preview" button may be provided which transiently displays the NIM on the user's computer screen without adding the NIM to the user's local processed user profile, as discussed above in relation to the N1MLink. All the NIM definitions that the user has listed on their home NIM are saved to the user profile database either periodically, at a set time, by event, or when the user closes their home NIM. The technique of the invention facilitates a viral distribution architecture. In other words, the technique of the invention facilitates rampant distribution of generated NIMs, as described below.

(205) Users (or developers) may share NIMs they have collected, and perhaps even modified, with other users (or developers) in accordance with this viral distribution architecture. Because the NIM definition contains basic reference information, such as data to instantiate the NIM and URLs and other references to where the NIM content is located, a NIM is easily and quickly distributed, collected, and shared. By-packaging Internet content and applications as NIMs and referencing the NIMs by NIMLinks, the system advantageously gives Internet content viral characteristics as the NIMs can easily be distributed or shared between users.

(206) Each NIM definition contains just enough information to define and initialize the NIM's components (NIM frame, controls, etc.). For example, this information may contain data to

configure the skeleton or frame that is filled in by NIM content from a developer's server. The NIM definition is therefore fairly small in size (-2K), and is therefore easily distributable as an XML file or Blob (binary large object), which is communicated using the same mechanisms (HTTP/HTTPS requests) as regular Web pages.

(207) This is especially useful where a user has collected a NIM or a group of NIMs (Packs) that he would like to send to another user. For example, a user may have an online trading NIM, calculator NIM, and stock research MM all set up in various positions on his screen, and would like to share the entire Pack with a friend who is remotely connected to the Internet.

(208) To share NIMs with others, the system utilizes the Sharelink database 78 of 25 FIG. 15 and the Sharing procedure 405 of FIG. 15. FIG. 24 is a diagrammatic illustration of the Sharelink database 78. NIM Sharelink database 78 stores a list of all NIMs shared by users, developers, or administrators, in a share table 484. Each NIM or group of NIMs shared is assigned a Share ID 486 which points to a Share module 488. Each Share Module 488 may also include a creation date 490, multiple 30 individual MM definition modules 492, or multiple packs of NIMs that have been shared (Sharepack module 494) containing multiple NIM definition modules 496 and 498.

(209) FIG. 25 is an illustration of a Share NIM's GUI 500. All dots collected by the user (466 of FIG. 21) can be shared with other users by clicking on the "Share" button 502 shown in FIG. 21. Once the user has clicked on the "Share" button 502, the GUI 500 shown in FIG. 25 is launched. The user may then highlight any of the NIMs or packs of NIMs 504 he has collected or created and thereafter share the NIMs or packs of NIMs 504 by clicking on the

"Share via email" button 506. It should be noted that other means of distributing the NIMs may be used together with, or instead of, email.

(210) When users share NIMs or NIM packs, their home NIM application generates a 10 share module, which may for example be an XML Blob containing the N1M definition or Sharepack modules shared. The shared NIM XML is then sent to, and saved in, the Sharelink database (78 of FIGS. 1 and 24). The Sharing procedure 405 of FIG. 15 then automatically generates a shared link (ShareLink) that references or points to the address of the shared XML on the Sharelink database. This ShareLink is then sent or 15 distributed (via email or posted on a Web site) to other users.

(211) If a user receives shared NIM(s) or pack(s) and has a home NIM installed on his client computer, then clicking on the Sharelink adds the NIM(s) to the user's home NIM and opens the shared NIM(s) on the user's screen. If a recipient of a shared NIM does not have the home NIM installed on his computer, then the home NIM is downloaded and installed (with the user's cooperation), the shared MM is added to his local processed user profile, and the NIM is opened.

(212) The NIM management module (112 of FIG. 15) may also be responsible for controlling and managing the development of new NIMs via the DevZone and the DevNIM discussed below.

(213) Because NIM content is based on existing Internet content standards (HTML, DHTML, GIFs, etc.) developers can create MM content using their existing Internet content development tools and methodologies. Therefore, no special hardware or software is required to develop or serve NIM content.

(214) Furthermore, as the application server (50 of FIG. 1) hosts and delivers NIM definitions from the NIM Templates database (74 of FIG. 1) developers merely define and package the NIM content without directly authoring, hosting, or serving the XML NIM definitions. Therefore, no special hardware or software is required on the developer-side to host and serve the NIM content, other than required for their regular Internet content.

(215) Two means are provided for creating NIMs. First, a Developer Zone Web site (DevZone) and second, a set of developing NIMs (DevNIMs). Both means enable NIM developers to create, define, and modify NIM definitions, and to support the NIM development process which results in XML NIM definitions being added to the NIM Templates database and NIMLinks generated.

(216) The DevZone is a Web site where NIM developers can view a list of NIMs they have defined and/or published, add new NIMs, and categorize, view, modify, or delete their existing NIMs. The DevZone is preferably rendered in a Web browser, is hosted on the Web server (82 of FIG. 1), and is implemented with a DevZone procedure (406 of FIG. 15). To access the DevZone, the developer may typically pass through a secure portal, such as by supplying a login identity and password.

(217) FIG. 26 is an illustration of the main DevZone Web page 510. All NIMs created by the NIM developer appear in a customized NIM list 514 that may only be accessed by that NIM developer. All NIMs created by a developer appear on the NIM list 514, unless they have been deleted by the developer or by a system administrator. The NIM list may contain the NIM name 518, the date the NIM was created 520, and an indication 516 of whether the NIM is in development or accessible by the public in the NIMIndex (i.e. "in-development" or "published").

(218) To access the NIM definition (for modification or review) the developer clicks on a "modify" or "preview" link 524 as transient (e.g. to add the NIM to their home NIM for previewing and testing). By clicking on the "modify" link, the developer is taken to the NIM modification web page, as shown in FIG. 27. Alternatively, by clicking on the "Create a Dot'" button 522, the developer is taken to a web page similar to the NIM modification web page shown in FIG. 27, where the developer may create a new NIM.

(219) FIG. 27 is a partial view of a NIM modification web page 530. To modify an existing NIM, or create a new NIM definition, a developer preferably utilizes Web forms, such as 534 to 542, or any area that contains objects that capture user input, such as text entry spaces, check boxes, and selection buttons. Developers typically fill in forms with information which defines the NIM, where the details might include the NIM's name 534, the URL for any image associated with NIM 540 (as shown in FIG. 18), the URL for a detailed image 542 (448 of FIG. 19), and such details as NIM frame (e.g., size of NIM, sizeable), layout of the controls (e.g., WebConduit control), and to specify the initial MM content (e.g., the initial target URLs for the WebConduit control, TitleBar, BottomBar), and any categories in which the developer would like the NIM to be listed in the NIMIndex. Once the developer has completed or modified the forms, he may either save or delete the NIM 532. If the developer selects either the development check box 536 or the public check box 538, and then saves the NIM, the DevZone procedure (407 of FIG. 15) generates a XML NIM definition, stores the XML NIM definition in the NIM Templates database (74 of FIGS. 1 and 15) and returns a NIMLink pointing to that NIM which is listed on the NIM list (514 of FIG. 26) on the developer's home NIM. The only difference being that once the developer selects the public check box 538 and saves the NIM, the NIM definition is

published utilizing the PubZone publishing procedure (410 of FIG. 15) to a publically accessible portion of the NIM Template Table (415 of FIG. 16), from where users can access, download, and collect the NIM. If the developer selects the development check box 536, the NIM can only be viewed and or modified by the developer and system administrator. It should be noted that the DevZone only allows control of certain characteristics of each NIM. Other characteristics may be set to default while still other characteristics can only be altered by an administrator. In an alternative embodiment, the DevZone may be rendered in a NIM or group of NIMs just as it was rendered in a Web browser. In either embodiment, NIM developers fill out one or more forms specifying NIM definition parameters, an XML NIM definition gets created and stored in the NIM Templates. Database, and a NIMLink gets generated that points to the new NIM. The Developer can then view or debug this NIM by clicking on the NIMLink to add it to his home NIM, or preview as transient, and thereafter render it on his screen. NIM definitions may also be developed using NIMs and NMA messages. A 30 developer may create Raw NIMs from empty NIM Templates using a development NIM (the DevNIM) on the developer's home NIM.

(220) FIG. 28A to 28D are GUIs of a development NIM (the Dev NIM). A developer may obtain a DevNIM by either collecting the DevNIM in the usual manner, as discussed above, or the system, via the system administrator, may share the NIM with the developer, also as discussed above. The DevNim contains a DevNIM procedure (408 of FIG. 15) which is transmitted to the developer's home NIM, as discussed above.

(221) To create a new NIM, the developer launches the DevNIM and enters a NIM name 550 into the DevNIM. The server then obtains an empty NIM (a NIM with default or no

initialization data and with only basic characteristics) from the NIM Template Database using the procedure for collecting NIMs described above, and saves the empty NIM under the supplied new NIM name 550 locally in the developer's processed user profile. The developer may then modify the empty NIM to the required form using the DevNIM. In the preferred embodiment a pull down menu 552 is provided where the developer can select which feature to modify, such as the frame characteristics (FIG. 28B), the titlebar (FIG. 28C), or initialization URLs for different frame or control elements (FIG. 28D).

(222) Each time the developer modifies a setting, the DevNIM, using the DevNIM procedure, sends NMA messages to the newly saved NIM to modify its definition parameters. For example, modifying the NIM's name, size, TitleBars, BottomBars, or WebConduits (as shown in FIGS. 28A to 28C).

(223) Unlike the DevZone, a new XML NIM definition and NIMLink is not generated every time a modification is made. All modifications (during the development cycle) are made locally to the NIM definition and are stored in the developer's processed user profile. The DevNIM embodiment, therefore, requires a separate publishing step that promotes the newly created NIM definition from the developer's user profile, to the NIM templates database on the application server.

(224) To publish the NIM, the NIM developer categorizes the NIM and the NIM definition is copied from the developer's processed user profile to the NIM Templates database.

(225) A publishing NIM (PubNIM), implemented with Publishing procedures 409, is provided to handle these functions. The PubNIM may therefore be shared or transmitted to the developer along with the DevNIM. The PubNIM contains a PubNIM procedure (411 of FIG. 15) which controls the publication of the NIMs to the NIM template database, as

discussed above. The PubNIM procedure sends a NIM definition module to the application server which receives the NIM definition module, extracts the NIM definition from the share module, stores it in the NIM Templates database, and associates the NIM with the developer so that the NIMLink shows up on the developer's NIM list (in their DevZone account).

(226) Alternatively, the new NIM may be published directly from the DevNIM. Once a user is satisfied with the NIM, he may select an option which publishes that NIM definition to the NIM Templates database. It should be noted that a developer may modify his NIMs at any time from the DevNIM.

(227) As mentioned earlier, when a developer is first authorized to create and/or modify NIMs, or at any time thereafter, information about that developer is saved in that developer's user profile (76 of FIG. 1) on the application server.

(228) A developer may also create application programs using NIMs, which a user may access from his client computer. Just as client-side application characteristics (sizing, position, menus) are accessible to content via NMA, the system may offer server-side application functionality, or toolkits, which are accessible through the NMA.

(229) A developer can build a NIM application without implementing, hosting, or supporting complex server or client applications. By using the server toolkits, a developer can develop N1Ms that exhibit server-application behavior by focusing on implementing NIM content (just like standard Internet content).

(230) For example, a NIM's content (an HTML page) may send a message to the system (or server) to request a credit card to be processed. Other toolkit examples may include credit

card billing, user profiling, targeted advertising, email, chat rooms, Internet telephony applications, or calendars.

(231) Any server-side application can be made accessible through the NMA, as a toolkit, just as client-side application behaviors are made accessible. In the current implementation, server-side application functions could be offered by a NIM (exposed via javascript functions on a page in a hidden frame). Other NIMs could access this functionality by sending NMA messages to this "Toolkit NIM" calling the functions. A NIM developer may therefore focus on Internet content development while accessing the features, behaviors, and functionality of an application just as if he had developed custom client and server side applications. The NIM management module (112 of FIG. 15) may also be responsible for controlling and managing the administration of the system via the AdminZone and the AdminNIM discussed below.

(232) A system administrator has the power to create, modify or delete users, developers, NIMs, other administrators, or NIMIndex categories, depending on that administrators access privileges. In a similar manner to the DevZone and DevNIM, system administrators may utilize either a Web browser administration zone (AdminZone), or an administration NIM (AdminNIM) which both make use of Administration procedures (412 of FIG. 15).

(233) To access the AdminZone, an administrator typically passes through a secure portal, such as by supplying a login identity and password. Once within the AdminZone, the administrator may search for a NIM by NIM name or title 552, category, developer, developer contact name, or status, as shown in FIG. 29A. The administrator may also selectively search for NIM's in development or publically accessible NIMs 554.

(234) Utilizing an AdminZone procedure (413 of FIG. 15), once the required NIM 20 is located the administrator may modify or delete the NIM in a similar manner to a developer as shown in FIG. 29B, and described above.

(235) Also utilizing the AdminZone procedure, the administrator may manage NIMIndex categories by creating new categories, modifying or deleting existing categories, and/or adjusting the layout of the NIMs within those categories as shown in FIGS. 30A and 30B. For example, an administrator may change a category's name 558, designate the category active or inactive 556, or create sub-categories 560. The system administrator may also select a category or categories for the NIM to appear in, where each NIM may be registered in more than one category.

(236) Finally, utilizing the AdminZone procedure (413 of FIG. 15), an administrator may search for users, providers, or developers and adjust their details, as shown in FIGS. 31A and 31 B. The system administrator may, for example, change a users contact details. In addition to adding, modifying or deleting NIMs, system administrator may have the task of reviewing NIM submissions from developers and promoting NIMs to the public. A submissions list of newly submitted NIMs may be displayed to an administrator, who may promote the NIM to the public or view the NIM. Once promoted, changes are made to the NIM Templates database and the NIM is automatically removed from the submissions list (again by utilizing the AdminZone procedure (413 of FIG. 15)).

(237) The foregoing discussion has explored the inherent nature of NIMs. Attention now turns to different techniques that may be used to exploit information that is associated with the use of NIMs. In particular, the following discussion is directed toward the accumulation

of statistical information that is only available in view of the architecture of the present invention.

(238) Currently, the predominant method of tracking and collecting user online behavior is severely limited for a number of reasons. First, most Internet use or visitor statistics are single-dimensional (linear, sequential) because Internet content is presented to users one full-screen page at a time. Second, users visit and leave sites so rapidly their visits are barely meaningful. Third, user's browsing habits are often discontinuous (browsers give users navigational bypass controls--back, forward, home, refresh, stop, etc.). Fourth, user behavior tracking is limited from a single site's server point of view. Current use statistics are plagued with the challenge of tracking continuous user behavior (especially from a cross-company perspective), with more than a single dimension of use context. Finally, because a computer may have multiple users, or a single user may use multiple computers, tracking continuous user on-line behavior is extremely difficult.

(239) One of the advantages of the NIM system as illustrated in FIG. 1, is that the Server 50 is able to track continuous, long-term NIM use information about each user. This is because the NIM server, through communication with the home NIM, can track each NIM event performed by each user. Therefore, it is possible to track each individual user's entire NIM use activity from the moment the user downloads the home NIM.

(240) Referring to FIG. 32, in one embodiment of the invention the following events may be tracked by the Event Log Module 98 (within the client computer 20 of FIG. 1): home NIM Download Event 704 NIM Download Event 706 NIM Display Event 708 Web Click-Through Event 710 Page-View Event 712 First NIM Installation Event 714 First

home NIM Startup Event 716 Transient Mode Event 718 Share NIM Received Event 720 NIM Pack Received Event 722.

(241) A home NIM Download Event 704 is logged when the user clicks on a link to request the home NIM user application. Preferably, the start time 752, and the end time 754 are recorded for this event. Also recorded is the provider ID 750 which is a parameter (generally, an integer) that represents the content provider partner who provided the link to the user.

(242) NIM Download Event 706 is logged when the home NIM acquires a NIM via a NIMLink. The start time 752, the end time 754, and the provider ID 750 are recorded for this event. Also recorded is the NIM ID 756 which is a parameter (generally, an integer) that represents the NIM that was just downloaded.

(243) NIM Display Event 708 is logged when a user activates a NIM. The NIM ID 756, the start time 752, and the end time 754 are recorded for this event. Web Click-Through Event 710 is logged whenever a user links from a NIM to a full-screen browser. This can occur when a user clicks on a link in the NIM, or it can occur automatically through the NIM messaging, or directly through the content provider. The NEM ID 756, the start time 752, and the Internet address 758 of the link are recorded. Page-View Event 712 is logged whenever a user views a page of content within a NIM. The NIM ID 756, and the start time 752 are recorded for this event.

(244) First NIM Installation Event 714 is logged the first, a NIM or NIM Pack is installed from a web site. This event is logged only once for each user account. The NIM ID 756, start time 752, and end time 754 are recorded for this event.

(245) First home NIM Startup Event 716 is logged when the home NIM runs for the first time. This event is logged only once for each user account. The start time 752 is recorded for this event.

(246) Transient Mode Event 718 is logged when the home NIM runs in transient mode. Transient mode occurs when the home NIM runs before the user has logged in. The start time 752, and the end time 754 are recorded for this event.

(247) Shared NIM Received Event 720 is logged for each NIM a user receives as part of a share. If a NIM Pack is shared, this event will be recorded for each NIM in the shared pack. The NIM ID 756, the start time 752, and the end time 754 are recorded for this event.

(248) Shared NIM Pack Received Event 722 is logged for each NIM Pack a user receives as part of a share. Thus, when a NIM Pack is shared, an Event 720 will be logged for each NIM in the NIM Pack, while an Event 722 will be logged once for the NIM Pack itself. The start time 752, and the end time 754 are recorded for this event.

(249) The events listed above are tracked in one particular embodiment. Other embodiments may track more or perhaps fewer events. This comprehensive event tracking is possible because each user event can be identified by the NIM Server through communication with the home NIM. Additional events may include tracking when a user sends a share or tracking when a user sends a NIM or a NIM Pack.

(250) FIG. 33 shows a typical series of user actions 800 as they are tracked by the Event Log Module 98. First, a user may request to download the home NIM application (step 802) from either a partner's web site or the NIM Server 50. The Event Log Module 98 records a home NIM Download Event, as shown with field 704 of FIG. 32. The start time 752, and

the end time 754 are preferably recorded. Also, the provider ID 750 of the site from where the home NIM download request was received is recorded.

(251) Returning to FIG. 33, the user subsequently activates the home NIM for the first time (step 804). The Event Log Module 98 records a First home NIM Startup Event 716, as shown in FIG. 32. The start time 752 is preferably recorded. In addition, the home NIM is activated and the user lot yet logged in, a Transient Mode Event 718 is logged and the start time 752 is recorded.

(252) As shown in step 806 of FIG. 33, the user logs into the home NIM. When this occurs, the end time 754 may be recorded for the Transient Mode Event 718.

(253) A user download of a new NIM (step 808) may be from a partner's web site or the NIM Server. When this occurs, the raw NIM definition is copied into the user's User Profile 76. The event log 98 records two events. First, because this is the first NIM the user has installed, a First NIM Installation Event 714 is recorded. The start time 752, the end time 754, and the provider ID 750 of the download site are preferably recorded. The second event recorded is a NIM Download Event 706. The Event Log Module 98 preferably tracks the NIM ID 756, the provider ID 750, the start time 752, and the end time 754 for this event. The next thing a user may do is open the NIM (step 810). This consists of retrieving the NIM definition from the user's User Profile and getting NIM content from the provider 82, as discussed above. The NIM is displayed for the user and the Event Log Module 98 records a NIM Display Event 708. However, at this point, the Event Log Module 98 can only record the start time 752, and the NIM ID 756 for this event. The end time 754 is recorded when the NIM is closed.

(254) For every page of content a user views within a NIM 812, a Page-View Event 712 is recorded. Some page views may require content from the provider 82. The NIM ID 756, and the start time 752 are recorded for this event.

(255) The NIM may also enable the user to click on a link that results in navigating to a full screen web browser (step 814). When a user does this, a Web Click-Through Event 710 is recorded. The Event Log Module 98 records the NIM ID 756, the start time 752, and the URL of the web site that is passed from the NIM content to the browser 758.

(256) When the NIM closes (step 816), the end time 754 for the NIM Display Event 708 is recorded. When the user logs out of the home NIM (step 818), the event log is uploaded to the Server 50 (of FIG. 1).

(257) In one embodiment of the invention, the previously described Event Log Module 98 (within the client computer 20 of FIG. 1) tracks user events in the home NIM user application and uploads the information to the Statistics Database 80 (of the server computer 50 of FIG. 1) at predetermined intervals alternate embodiments, the Event Log 700 (in FIG. 32) may be processed by the NIM Server before it is stored in the Statistics Database 80. For example, the NIM Server may process NIM use status information for each user that is currently logged in.

(258) The Statistics Database 80, illustrated in FIG. 34, preferably lists every event 1002 by every user of home NIMs along with the corresponding fields associated with each event. For-example, if a NIM Display Event is recorded, the User ID 1004 of the user that performed the event is listed, the start time 1006 is listed, the end time 1008 is listed, and the NIM ID 1010 is listed. If a Web Click-Through Event is recorded, the User ID 1004 is listed, the NIM ID 1010 is listed, the start time 1006 is listed, and the URL of the web site

1014 is listed. The Statistics Database 80 therefore allows the list of events to be easily referenced and searched by each event or by each of the fields associated with the events.

(259) Referring to FIG. 35, the Statistical Analysis Module 900 uses the Statistics Database 80 in order to provide various services for the content provider partners 82. Preferably, the Statistical Analysis Module 900 includes a Multi-Dimensional Consumer Profile Module 902, a Real-Time Advertising Module 904, and a Pack Building Module 906, as discussed below.

(260) A primary advantage of the present invention is that, because NIMs are used in groups and are used more often and for longer periods of time than web pages or web sites, real-time multi-dimensional NIM use data (that's a function of which NIMs are activated simultaneously) can be accumulated. In accordance with an embodiment of the invention, this accumulated data is used to generate a multi-dimensional consumer profiling database. The Multi-Dimensional Consumer Profile Module 902 uses information from the Statistics Database 80 to examine, for each user, the start time, and the end time of each NIM Display Event It then determines the NIMs (using the NIM IDs) that are opened simultaneously for each user. The Module 902 determines, for every selected NIM, the other NIMs that a given user may use in conjunction with the selected NIM. The Module 902 also determines how often these other NIMs are used simultaneously with the selected NIM. For example, Company X provides a NIM for selling its books. The Multi-Dimensional Consumer Profile Module 902 determines for Company X that a particular user has a NIM related to finance activated 30% of the time the user has the book-selling NIM acted, a NIM related to computers 20% of the time the user has the book-selling NIM activated, and a NIM related

to wedding gifts 5% of the time the user has the book-selling NIM activated. This will provide Company X with a more complete profile of the user's interests.

(261) The Real-Time Advertising Module 904 determines the NIMs that each user has displayed at any given moment. This information is used by a content provider partner or by the NIM Server to target advertising information. For example, if a user has a NIM related to sports displayed simultaneously with Company X's book-selling NIM, Company X uses this information to stream an advertisement for a sports book. In one embodiment, this is accomplished by associating each NIM with a context keyword. This is done by incorporating the context keyword into the NIM definition or, alternatively, by maintaining a table of NIMs and their corresponding context keywords. For example, the NIM related to sports is associated with the context keyword "sports." Moreover, the Real-Time Advertising Module 904 may combine the real-time user information with the historical user information from the Statistics Database 80 to provide advertisers with a complete picture of a user's interests.

(262) The Pack Building Module 906 uses the Statistics Database 80 to determine which NIMs are being used simultaneously. The Module 906 also determines which NIMs are being shared as NIM Packs. From this, the Module 906 provides information to content provider partners about which NIMs should be bundled together. In alternate embodiments, the Module 906 builds a NIM Pack based upon the information it processes. For example, if the Pack Building Module 906 determines that an airline NIM is being used with a hotel NIM and a car rental NIM, the Module 906 may build a NIM Pack with a restaurant NIM.

(263) Additionally, in one embodiment of the present invention, the NIM Server 82 may track the content within a NIM in a Content Database 1050, as illustrated in FIG. 36. A content

descriptor 1052 which may be a string describing the content that is shown within the NIM is recorded for content shown in the NIM. For example, if a NIM displayed an advertisement for an automobile followed by an advertisement for a restaurant, the two recorded content descriptors might say "automobile ad" and "restaurant ad." In addition, the NIM ID 1054, the start time at which the content is displayed 1056, and the end time 1058 are all preferably recorded for each content descriptor.

(264) Referring to FIG. 35, The Content Analysis Module 950 is able to correlate, at any moment, the content displayed to the user as recorded in the Content Database with the user's NIM activity recorded in the Statistics Database. For example, if one NIM displays to a user an advertisement for a travel book, the user may open a NIM related to Florida, a NIM owned by a specific airline, and a NIM owned by a car rental company. This pattern of user behavior will allow the company that provides the travel book advertisement to better understand the effect of the advertisement on the user. The company may use this information to make cross-promotions with other NIM providers, or, simply to provide more effective targeted advertisements.

(265) In an alternative embodiment, each of the content providers may track its own content information. The content providers could then compare its content information with the user information provided by the Statistics Database of the NIMServer.

(266) Finally, referring to FIG. 37, all of the user event information may be used in conjunction with user information provided at login. During the login process, the user may be required to enter demographic information such as age, marital status, etc. In one embodiment, this information is stored in a User Account Database 1100. Each User ID 1102 is listed along with the corresponding user information 1104. Therefore, it is possible

to match the user events with personal information about the particular user to give advertisers or NIM content providers a more complete behavior profile of each user.

(267) The foregoing description, for purposes of explanation, used specific nomenclature to provide a thorough understanding of the invention. However, it will be apparent to one skilled in the art that the specific details are not required in order to practice the invention. In other instances, well known circuits and devices are shown in block diagram form in order to avoid unnecessary distraction from the underlying invention. Thus, the foregoing descriptions of specific embodiments of the present invention are presented for purposes of illustration and description. They are not intended to be exhaustive or to limit the invention to the precise forms disclosed, obviously many modifications and variations are possible in view of the above teachings. The embodiments were chosen and described in order to best explain the principles of the invention and its practical applications, to thereby enable others skilled in the art to best utilize the invention and various embodiments with various modifications as are suited to the particular use contemplated. It is intended that the scope of the invention be defined by the following claims and their equivalents.

Electronic Acknowledgement Receipt

EFS ID:	15126253
Application Number:	11932553
International Application Number:	
Confirmation Number:	2604
Title of Invention:	Displaying Time-Varying Internet Based Data Using Application Media Packages
First Named Inventor/Applicant Name:	John Albert Kembel
Customer Number:	909
Filer:	D. Benjamin Esplin/Maxine Cutietta
Filer Authorized By:	D. Benjamin Esplin
Attorney Docket Number:	043311-0381668
Receipt Date:	05-MAR-2013
Filing Date:	31-OCT-2007
Time Stamp:	18:25:47
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Specification	0381668_SubSpecCLEAN_Filed 03052013.pdf	395163 135cdc59400cd52e84abc8adf48c30d661801bc8	no	92
Warnings:					
Information:					

2	Specification	0381668_SubSpecMARKEDUP_Filed03052013.pdf	1123853 afa193eff9629cf9acea067927107dd61cad63b0	no	248

Warnings:	
Information:	
Total Files Size (in bytes):	1519016

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Attorney Docket: 043311-0381668

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE PATENT APPLICATION OF	:	JOHN ALBERT KEMBEL, *et al.*
SERIAL NO.	:	11/932,553
FILING DATE	:	October 31, 2007
ART UNIT	:	2478
EXAMINER	:	K.S. LIN
FOR	:	DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES

RESPONSE

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

In response to the Office Action mailed **September 4, 2012** ("Office Action"), please consider the following amendments and remarks in the above-identified application:

Amendments to the Specification begin on page **2** of this paper.

Amendments to the Drawings begin on page **3** of this paper.

Amendments to the Claims are reflected in the listing of claims beginning on page **4** of this paper.

Remarks/Arguments begin on page **11** of this paper.

It is believed that no extensions of time or fees for net addition of claims are required beyond those that may otherwise be provided for in documents accompanying this paper. However, if additional extensions of time are necessary to prevent abandonment of this application, then such extensions of time are hereby petitioned for under 37 C.F.R. § 1.136(a), and any fees required therefore (including fees for net addition of claims) are hereby authorized to be charged to our Deposit Account No. 033975 (Ref. No. 043311-0381668).

AMENDMENTS TO THE SPECIFICATION

Please replace the specification as originally filed with the **SUBSTITUTE SPECIFICATION** filed herewith. In accordance with 37 C.F.R. § 1.125, a clean version of the **SUBSTITUTE SPECIFICATION** is filed herewith, along with a marked-up version showing the changes made.

AMENDMENTS TO THE DRAWINGS

Please replace sheets 1-14 of the drawings as originally filed with the **REPLACEMENT SHEETS** filed herewith.

CLAIM AMENDMENTS

This **listing of claims** will replace all prior versions, and listings, of claims in the application:

1. **(Currently Amended)** A client computing device configured to access content over a network, the client computing device comprising:

electronic storage configured to store networked information monitor ~~an application media package~~ template associated with a networked information monitor ~~an application media package~~, the networked information monitor ~~application media package~~ template having therein a definition of a viewer graphical user interface having a frame within which time-varying content in a web browser-readable language may be presented on a display associated with the client computing device, wherein the frame of the viewer graphical user interface lacks ~~user~~ controls ~~for~~ enabling a user to specify a network location at which content for the ~~manual network navigation with the~~ networked information monitor is available ~~application media package~~; and

one or more processors configured to execute one or more computer program modules, the one or more computer program modules being configured to access the networked information monitor ~~application media package~~ defined by the networked information monitor ~~application media package~~ template, wherein accessing the networked information monitor ~~application media package~~ defined by the networked information monitor ~~application media package~~ template results in:

transmission, over a network to a web server at a network location, of a content request for content to be displayed within the frame of the viewer graphical user interface defined by the networked information monitor ~~application media package~~ template;

reception, over the network from the web server at the network location, of content transmitted from the web server in response to the content request, the content being time-varying;

presentation, on the display, of the viewer graphical user interface defined by the networked information monitor ~~application media package~~ template~~, wherein the~~

~~lack of user controls for manually navigating a network with the application media package prevents manual user selection of the web server and prevents manual user selection of the network location~~ outside of and separate from any graphical user interface of any other application; and

presentation, on the display within the frame of the viewer graphical user interface defined by the networked information monitor ~~application media package~~, of the time-varying content received from the web server~~; and~~

~~responsive to reception of one or more elements included in the received time-varying content, modifying a feature of said graphical user interface defined by the application media package template in accordance with a modification corresponding to the received one or more elements~~.

2. **(Currently Amended)** The client computing device of claim 23 [[1]], wherein said modification corresponding to the received one or more elements comprises a modification to an image defined by the networked information monitor ~~application media package~~ template as forming a part of said graphical user interface.

3. **(Currently Amended)** The client computing device of claim 23 [[1]], wherein the correspondence between the modification and the received one or more elements is defined by the networked information monitor ~~application media package~~ template.

4. **(Currently Amended)** The client computing device of claim 23 [[1]], wherein the one or more computer program modules and the networked information monitor ~~application media package~~ template are configured such that modifying the feature of the viewer graphical user interface comprises adjusting a size of the [[a]] frame of the viewer graphical user interface.

5. **(Currently Amended)** The client computing device of claim 23 [[1]], wherein the one or more computer program modules and the networked information monitor ~~application media package~~ template are configured such that modifying the feature of the viewer graphical user interface comprises changing a color of a frame border or

background of the viewer graphical user interface.

6. **(Currently Amended)** The client computing device of claim 23 [[1]], wherein the one or more computer program modules and the networked information monitor ~~application media package~~ template are configured such that modifying the feature of the viewer graphical user interface comprises modifying text of the viewer graphical user interface in a manner defined by the networked information monitor ~~application media package~~ template.

7. **(Currently Amended)** The client computing device of claim 1, wherein the networked information monitor ~~application media package~~ template includes ~~is included within~~ a ~~single~~ markup language file.

8. **(Previously Presented)** The client computing device of claim 1, wherein one or more computer program modules are configured such that the time-varying content is received from the web server over the network according to the TCP/IP protocol.

9. **(Currently Amended)** The client computing device of claim 1, wherein the network location corresponds to a uniform resource locator included in the networked information monitor ~~application media package~~ template.

10. **(Currently Amended)** The client computing device of claim 9, wherein the one or more computer program modules are further configured such that accessing the networked information monitor ~~application media package~~ defined by the networked information monitor ~~application media package~~ template results in transmission of the content request to the uniform resource locator included in the networked information monitor ~~application media package~~ template, and the content request being transmitted according to the TCP/IP protocol over the network.

11. **(Currently Amended)** The client computing device of claim 1, wherein the one or more computer program modules are further configured:

to transmit, over the network to an networked information monitor ~~application media package~~ server, a request for the networked information monitor ~~application media package~~ template;

to receive, from the networked information monitor ~~application media package~~ server over the network, the networked information monitor ~~application media package~~ template; and

to store the networked information monitor ~~application media package~~ template to the electronic storage.

12. **(Currently Amended)** A computer-implemented method of access content over a network on a client computing device, the client computing device having electronic storage and one or more processors configured to execute one or more computer program modules, the client method comprising:

storing, to the electronic storage, a networked information monitor ~~an application media package~~ template associated with a networked information monitor ~~an application media package~~, the networked information monitor ~~application media package~~ template having therein a definition of a viewer graphical user interface having a frame within which time-varying content in a web browser-readable language may be presented on a display associated with the client computing device, wherein the frame of the viewer graphical user interface lacks ~~user~~ controls for enabling a user to specify a network location at which content for ~~manual network navigation with~~ the networked information monitor is available ~~application media package~~;

accessing the networked information monitor ~~application media package~~ defined by the networked information monitor ~~application media package~~ template, wherein accessing the networked information monitor ~~application media package~~ defined by the networked information monitor ~~application media package~~ template results in:

transmission, over a network to a web server at a network location, of a content request for content to be displayed in the viewer graphical user interface defined by the networked information monitor ~~application media package~~ template;

reception, over the network from the web server at the network location, of content transmitted from the web server in response to the content request, the content

being time-varying;

presentation, on the display, of the viewer graphical user interface defined by the application media package template~~, wherein the lack of user controls for manually navigating a network with the application media package prevents manual user selection of the web server and prevents manual user selection of the network location~~ outside of and separate from any graphical user interface of any other application; and

presentation, on the display within the frame of the viewer graphical user interface defined by the networked information monitor ~~application media package~~, of the time-varying content received from the web server~~; and~~

~~responsive to reception of one or more elements included in the received time-varying content, modifying a feature of said graphical user interface defined by the application media package template in accordance with a modification corresponding to the received one or more elements~~.

13. **(Currently Amended)** The method of claim 24 [[12]], wherein said modification corresponding to the received one or more elements comprises a modification to an image defined by the networked information monitor ~~application media package~~ template as forming a part of said viewer graphical user interface.

14. **(Currently Amended)** The method of claim 24 [[12]], wherein the correspondence between the modification and the received one or more elements is defined by the networked information monitor ~~application media package~~ template.

15. **(Currently Amended)** The method of claim 24 [[12]], wherein modifying the feature of the viewer graphical user interface comprises adjusting a size of the [[a]] frame of the viewer graphical user interface.

16. **(Currently Amended)** The method of claim 24 [[12]], wherein modifying the feature of the viewer graphical user interface comprises changing a color of a frame border or background of the viewer graphical user interface.

17. **(Currently Amended)** The method of claim 24 [[12]], wherein modifying the feature of the viewer graphical user interface comprises modifying text of the viewer graphical user interface in a manner defined by the networked information monitor ~~application media package~~ template.

18. **(Currently Amended)** The method of claim 12, wherein the networked information monitor ~~application media package~~ template includes ~~is included within~~ a ~~single~~ markup language file, and wherein storing the networked information monitor ~~application media package~~ template comprises storing the ~~single~~ markup language file.

19. **(Previously Presented)** The method of claim 12, wherein the time-varying content is received from the web server over the network according to the TCP/IP protocol.

20. **(Currently Amended)** The method of claim 12, wherein the network location corresponds to a uniform resource locator included in the networked information monitor ~~application media package~~ template.

21. **(Currently Amended)** The method of claim 20, wherein accessing the networked information monitor ~~application media package~~ defined by the networked information monitor ~~application media package~~ template results in transmission of the content request to the uniform resource locator included in the networked information monitor ~~application media package~~ template, and the content request being transmitted according to the TCP/IP protocol over the network.

22. **(Currently Amended)** The method of claim 12, further comprising:

prior to storing the networked information monitor ~~application media package~~ template to the electronic storage, transmitting, over the network to a networked information monitor ~~an application media package~~ server, a request for the networked information monitor ~~application media package~~ template; and

receiving, from the networked information monitor ~~application media package~~ server over the network, the ~~single markup language file including the~~ networked information monitor ~~application media package~~ template.

23. **(New)** The method of claim 1, further comprising, responsive to reception of one or more elements included in the received time-varying content, modifying a feature of said viewer graphical user interface defined by the networked information monitor template in accordance with a modification corresponding to the received one or more elements.

24. **(New)** The method of claim 12, responsive to reception of one or more elements included in the received time-varying content, modifying a feature of said viewer graphical user interface defined by the networked information monitor template in accordance with a modification corresponding to the received one or more elements

REMARKS

By this paper, claims 1-7, 9-18, and 20-22 are amended, claims 23-24 are added, and no claims are cancelled. Accordingly, claims 1-24 are all of the pending claims. Support for the amendments presented above is provided throughout the specification and claims as originally filed. Applicants expressly reserve the right to prosecute the subject matter of the unamended and/or cancelled claims, or any other subject matter supported by the Specification, in one or more continuation applications. In view of the foregoing amendments and the following remarks, reconsideration and allowance of all the pending claims is anticipated.

Specification

The amendments to the specification included in the **SUBSTITUTE SPECIFICATION** include replacing substantially all of the text in the specification as originally filed with the text of the specification of U.S. Patent No. 7,660,868 ("the '868 Patent"), from which the present application claims priority, and which was incorporated by reference at the time the present application was filed. As such, the inclusion of the text of the specification of the '868 Patent was supported by the present application at the time of filing.

Drawings

The amendments to the drawings included in the **REPLACEMENT SHEETS** include replacing the drawings as originally filed with the drawings of the '868 Patent, from which the present application claims priority, and which was incorporated by reference at the time the present application was filed. As such, inclusion of the drawings of the '868 Patent was supported by the present application at the time of filing.

Double Patenting

Claims 1-22 currently stand rejected on the ground of nonstautory obviousness-type double patenting as allegedly being unpatentable over each of (1) claims 1-32 of U.S. Patent Application No. 11/932,286, (2) claims 1-5, 7-12, 17, and 19-25 of U.S. Patent Application No. 11/932,392, (3) claims 1-25 of U.S. Patent Application No. 11/932,427, (4) claims 1-20 of U.S. Patent Application No. 11/932,630, and (5) claims 1-20 of U.S. Patent Application No. 11/932,692. Applicants traverse each of these rejections at least on the grounds that the claims of the cited applications do not teach or suggest all of the features of the claimed invention. Nevertheless, solely in an effort to expedite prosecution, Applicants will consider filing the appropriate Terminal Disclaimers should the present application be indicated as being otherwise in condition for allowance.

Claim Rejections – 35 USC § 103

Claims 1-5 and 7-22 currently stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over U.S. Patent No. 5,838,906 to Doyle *et al.* ("Doyle") in view of U.S. Patent No. 5,375,199 to Harrow *et al.* ("Harrow"). Applicants traverse this rejection at least on the grounds that the proposed combination of Doyle and Harrow does not teach or suggest all of the features of the claimed invention. However, solely in an effort to expedite prosecution, Applicants have amended the claims to further clarify various aspects of the claimed invention.

For example, claim 1, as amended, recites *inter alia* the following features, which are not taught or suggested in the proposed combination of Doyle and Harrow:

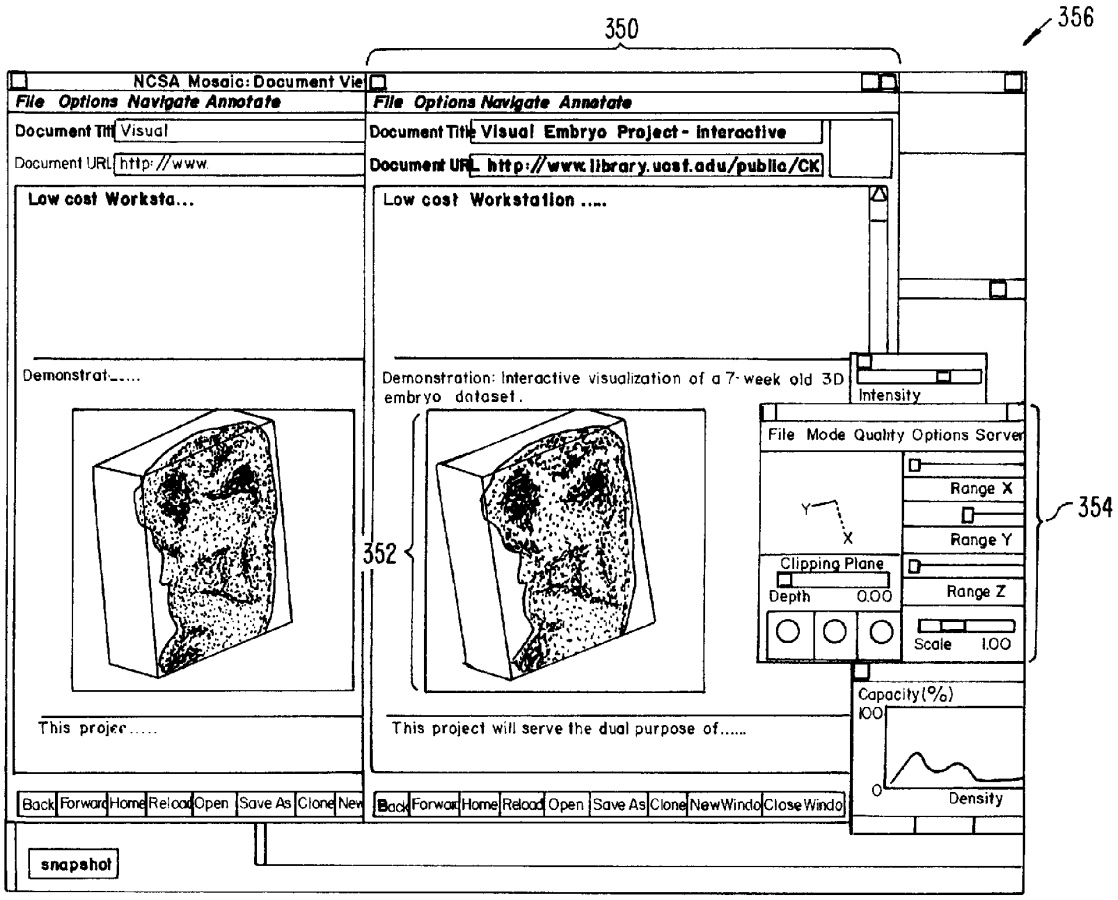
> …electronic storage configured to store a networked information monitor template… having therein a definition of a viewer graphical user interface…, **wherein the frame of the viewer graphical user interface lacks controls enabling a user to specify a network location at which content for the networked information monitor is available**; and
>
> one or more processors configured to execute one or

> more computer program modules… result[ing] in:
>
> …presentation, on the display, of the viewer graphical user interface defined by the networked information monitor template **outside of and separate from any graphical user interface of any other application**….

Independent claim 12, as amended, recites *inter alia* the following features:

> …storing… a networked information monitor template… having therein a definition of a graphical user interface…, **wherein the frame of the viewer graphical user interface lacks controls enabling a user to specify a network location at which content for the networked information monitor is available**; and
>
> accessing the networked information monitor… result[ing] in:
>
> …presentation, on the display, of the viewer graphical user interface defined by the networked information monitor template **outside of and separate from any graphical user interface of any other application**….

The cited sections of Doyle, for example at FIG. 9 (reproduced) below, appear to describe presenting a graphical user interface for one application (*e.g.*, one lacking controls in its frame for enabling a user to specify a network location at which content is available) within a web browser. As can be further seen in FIG. 9, the graphical user interface window of the web browser has traditional controls for enabling a user to specify a network location at which content is available for presentation in its frame.



At least because the web browser interface has a frame with controls that enable a user to specify a uniform resource locator, the web browser does not teach the networked information monitor recited in the claims. At least because the graphical user interface of the application that is shown within the web browser is not presented outside of and separate from any other graphical user interface of an application (*e.g.*, the graphical user interface of the web browser), the application shown within the web browser does not teach or suggest the networked information monitor recited in the claims. Therefore, the cited sections of Doyle fail to teach or suggest the features of claims 1 and 12 reproduced above.

The cited sections of Harrow teach displaying content within the user interface of a conventional web browser, such as Mosaic [*see*, *e.g.*, FIG 9, and c. 16, ll. 9-28].

Conventional web browsers have graphical user interfaces with frames that include controls which permit manual network navigation (hence the name "web browser"). As such, the sections of Harrow relied on in the Office Action do not address the deficiencies of Doyle set forth above with respect to claims 1 and 12.

Therefore, the cited sections of Doyle and Harrow, alone or in combination, fail to teach or suggest the features of claims 1 and 12 reproduced above. For at least this reason the rejection of claims 1 and 12 under § 103 should be withdrawn.

Claims 2-11 and 13-22 depend from a corresponding one of claims 1 and 12. As such, the rejection of claims 2-11 and 13-22 under § 103 should be withdrawn due to the dependency of these claims, as well as for the features that they recite individually.

Newly Added Claims

Claims 23 and 24 are added by this paper. Claims 23 and 24 depend from claims 1 and 12, respectively. As such, claims 23 and 24 are believed to be in condition for allowance by virtue of the dependency of their dependency, as well as for the features they recite individually.

CONCLUSION

Having addressed each of the foregoing objections and rejections, it is respectfully submitted that a full and complete response has been made to the outstanding Office Action and, as such, the application is in condition for allowance. Notice to that effect is anticipated.

If the Examiner believes, for any reason, that personal communication will expedite prosecution of this application, the Examiner is invited to telephone the undersigned at the number provided.

Date: March 4, 2013

Respectfully submitted,

By: /D. Benjamin Esplin/
D. Benjamin Esplin
Reg. No. 58,297

Customer No. 00909

PILLSBURY WINTHROP SHAW PITTMAN LLP
1650 Tysons Boulevard
McLean, Virginia 22102
619-234-5000

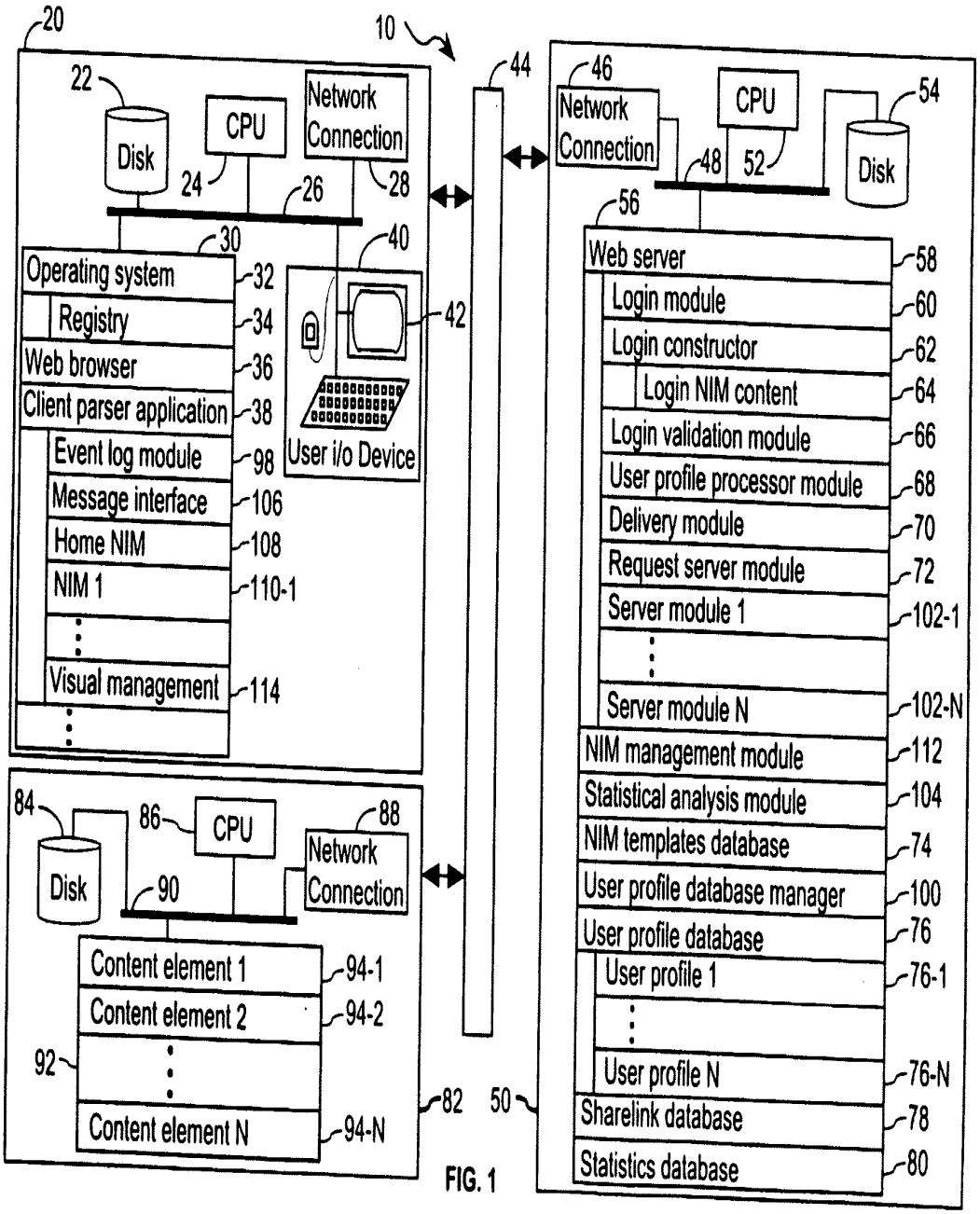

20
10
22
Disk
CPU
24
Network Connection
26
28
44
46
Network Connection
CPU
48
52
54
Disk
56
30
Operating system
32
Registry
34
Web browser
36
Client parser application
38
Event log module
98
Message interface
106
Home NIM
108
NIM 1
110-1
Visual management
114
40
42
User i/o Device
Web server
58
Login module
60
Login constructor
62
Login NIM content
64
Login validation module
66
User profile processor module
68
Delivery module
70
Request server module
72
Server module 1
102-1
Server module N
102-N
NIM management module
112
Statistical analysis module
104
NIM templates database
74
User profile database manager
100
User profile database
76
User profile 1
76-1
User profile N
76-N
Sharelink database
78
Statistics database
80
84
Disk
86
CPU
88
Network Connection
90
92
Content element 1
94-1
Content element 2
94-2
Content element N
94-N
82
50

FIG. 1

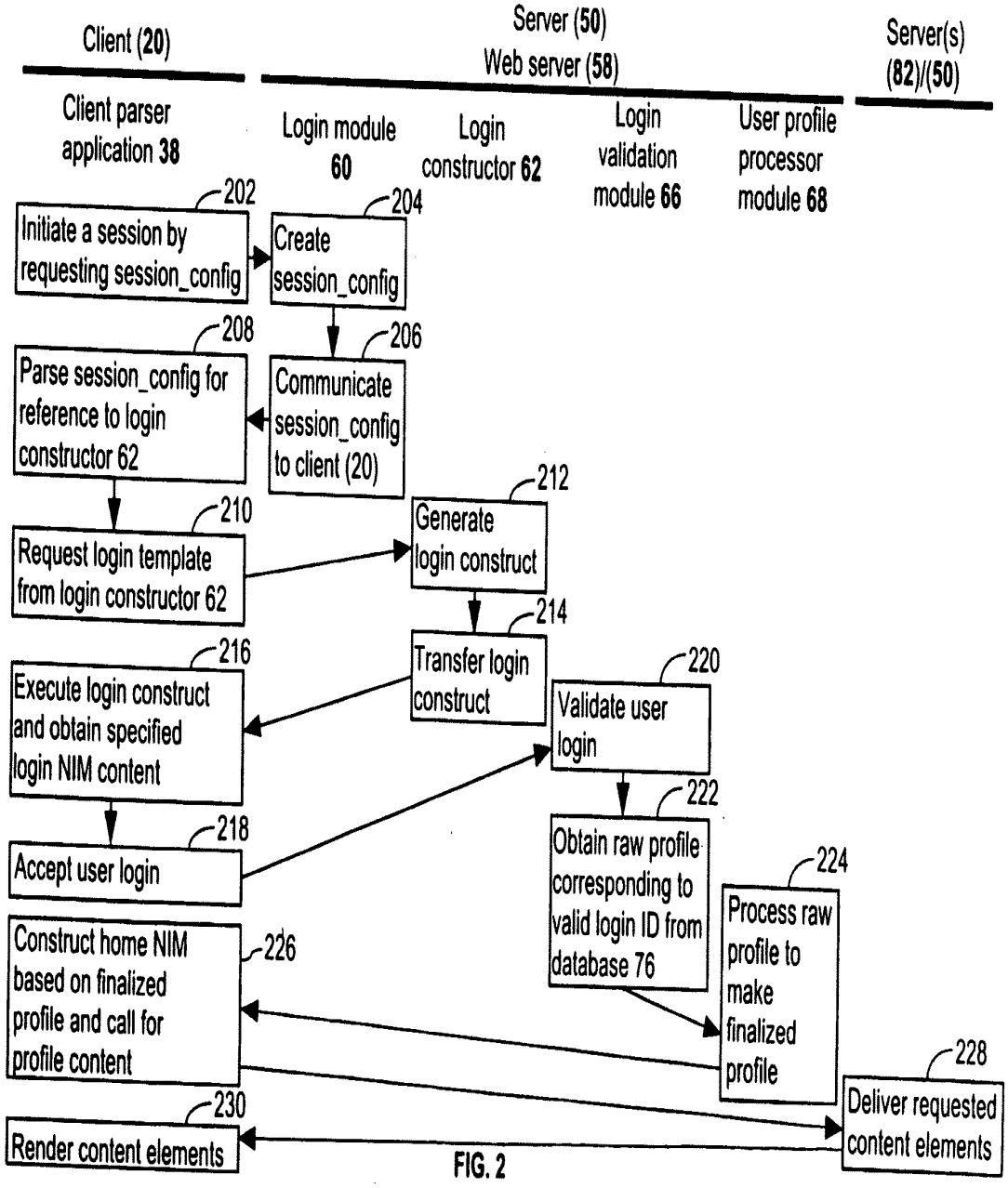

Client (20)
Server (50)
Web server (58)
Server(s) (82)/(50)
Client parser application 38
Login module 60
Login constructor 62
Login validation module 66
User profile processor module 68
202
Initiate a session by requesting session_config
204
Create session_config
208
Parse session_config for reference to login constructor 62
206
Communicate session_config to client (20)
210
Request login template from login constructor 62
212
Generate login construct
214
Transfer login construct
216
Execute login construct and obtain specified login NIM content
220
Validate user login
218
Accept user login
222
Obtain raw profile corresponding to valid login ID from database 76
224
Process raw profile to make finalized profile
226
Construct home NIM based on finalized profile and call for profile content
228
Deliver requested content elements
230
Render content elements

FIG. 2

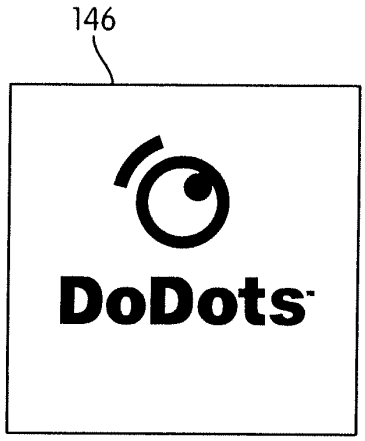

146
DoDots

FIG. 3A

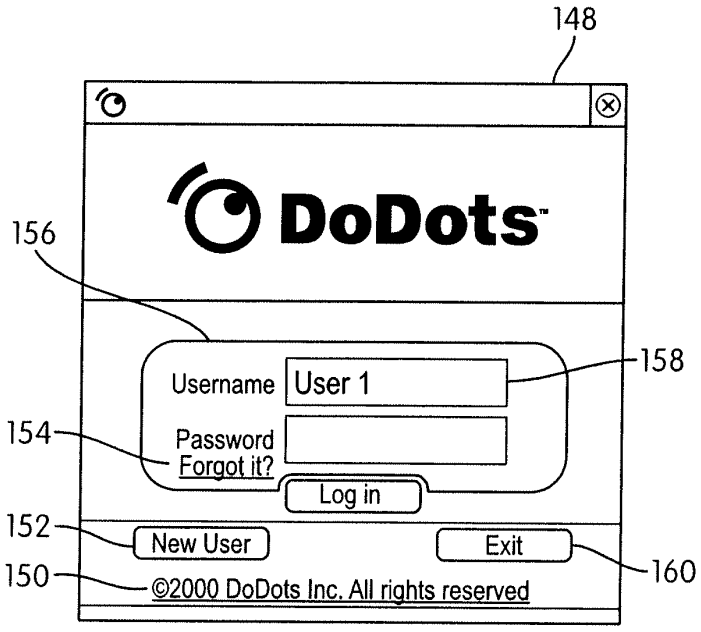

148
DoDots
156
Username
User 1
158
154
Password
Forgot it?
Log in
152
New User
Exit
160
150
©2000 DoDots Inc. All rights reserved

FIG. 3B

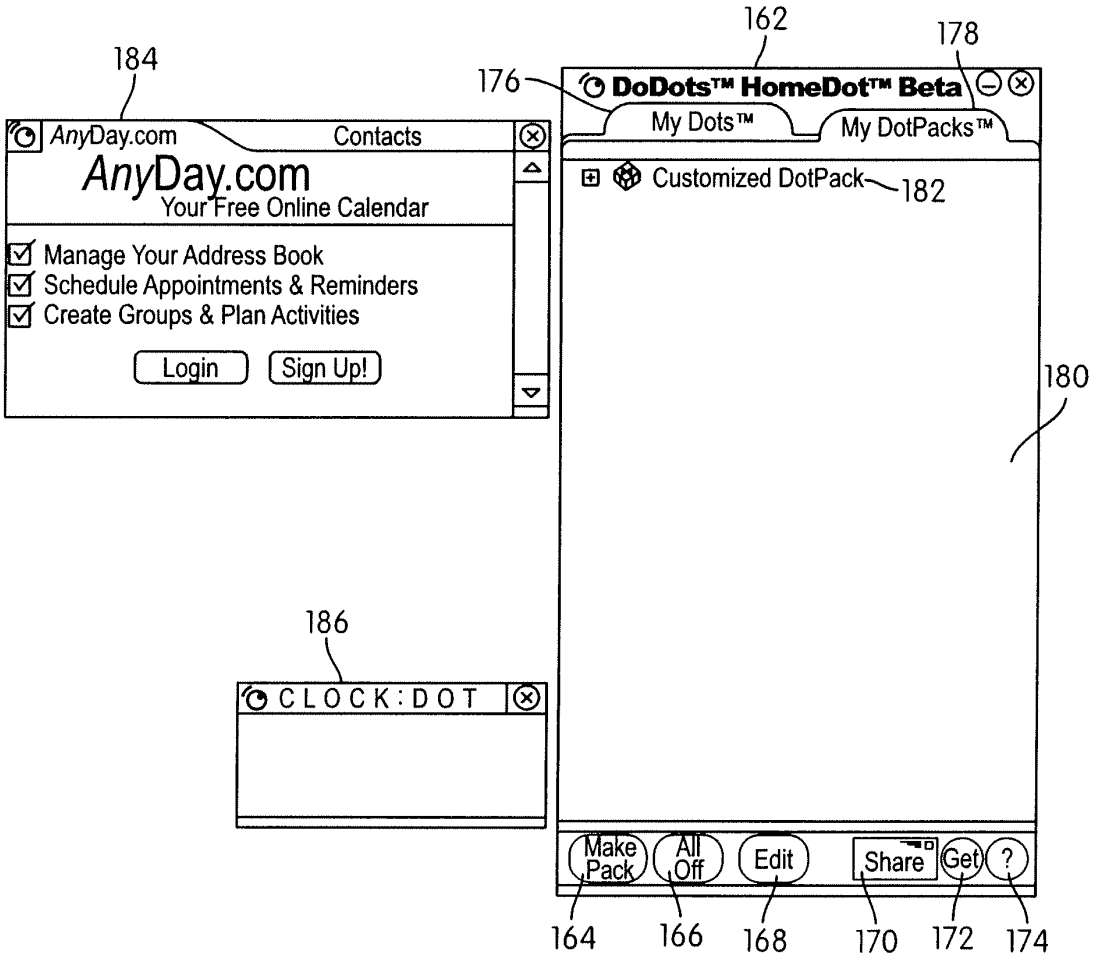

162
178
184
176
DoDots™ HomeDot™ Beta
My Dots™
My DotPacks™
AnyDay.com
Contacts
AnyDay.com
Your Free Online Calendar
Manage Your Address Book
Schedule Appointments & Reminders
Create Groups & Plan Activities
Login
Sign Up!
Customized DotPack
182
180
186
CLOCK:DOT
Make Pack
All Off
Edit
Share
Get
?
164
166
168
170
172
174

FIG. 4

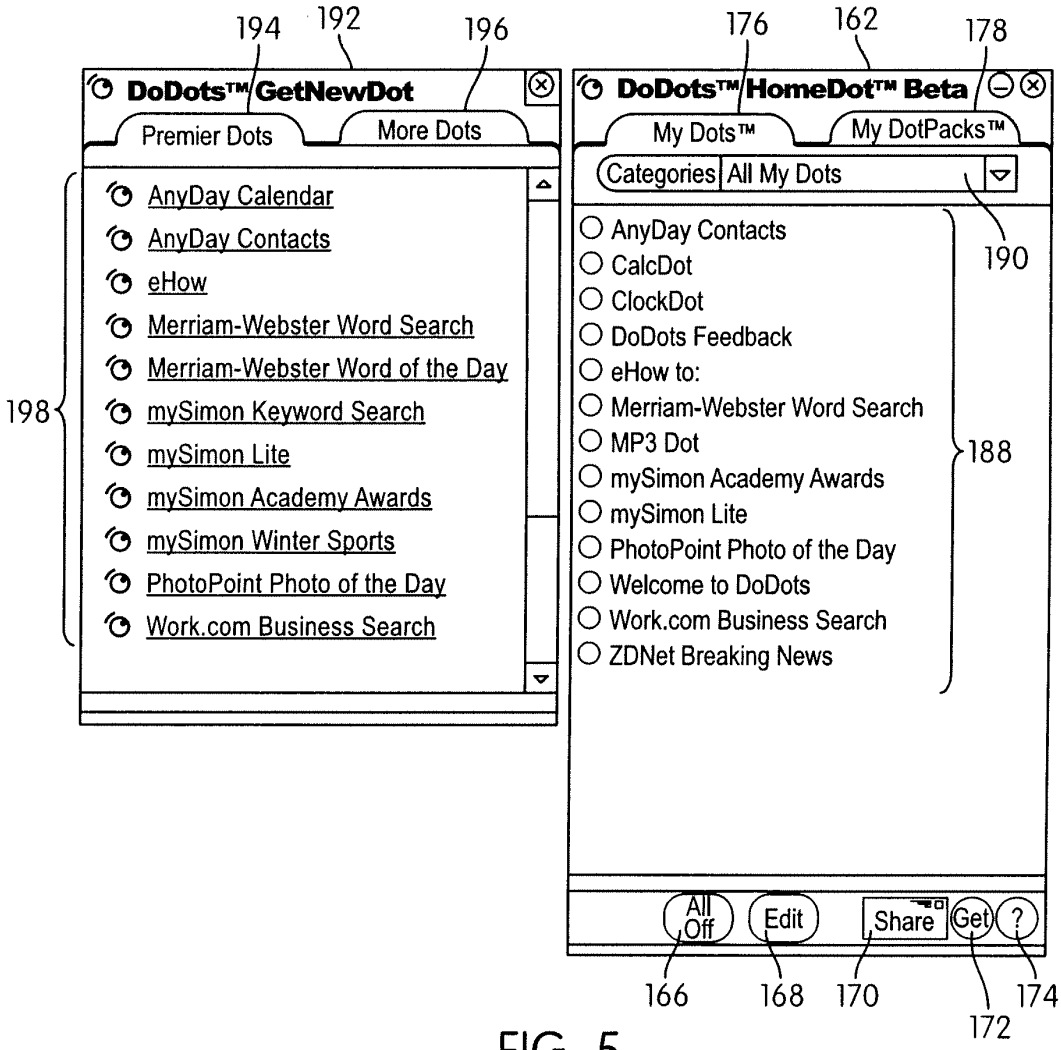

194
192
196
176
162
178
DoDots™ GetNewDot
Premier Dots
More Dots
AnyDay Calendar
AnyDay Contacts
eHow
Merriam-Webster Word Search
Merriam-Webster Word of the Day
mySimon Keyword Search
mySimon Lite
mySimon Academy Awards
mySimon Winter Sports
PhotoPoint Photo of the Day
Work.com Business Search
198
DoDots™ HomeDot™ Beta
My Dots™
My DotPacks™
Categories
All My Dots
190
AnyDay Contacts
CalcDot
ClockDot
DoDots Feedback
eHow to:
Merriam-Webster Word Search
MP3 Dot
mySimon Academy Awards
mySimon Lite
PhotoPoint Photo of the Day
Welcome to DoDots
Work.com Business Search
ZDNet Breaking News
188
All Off
Edit
Share
Get
?
166
168
170
172
174

FIG. 5

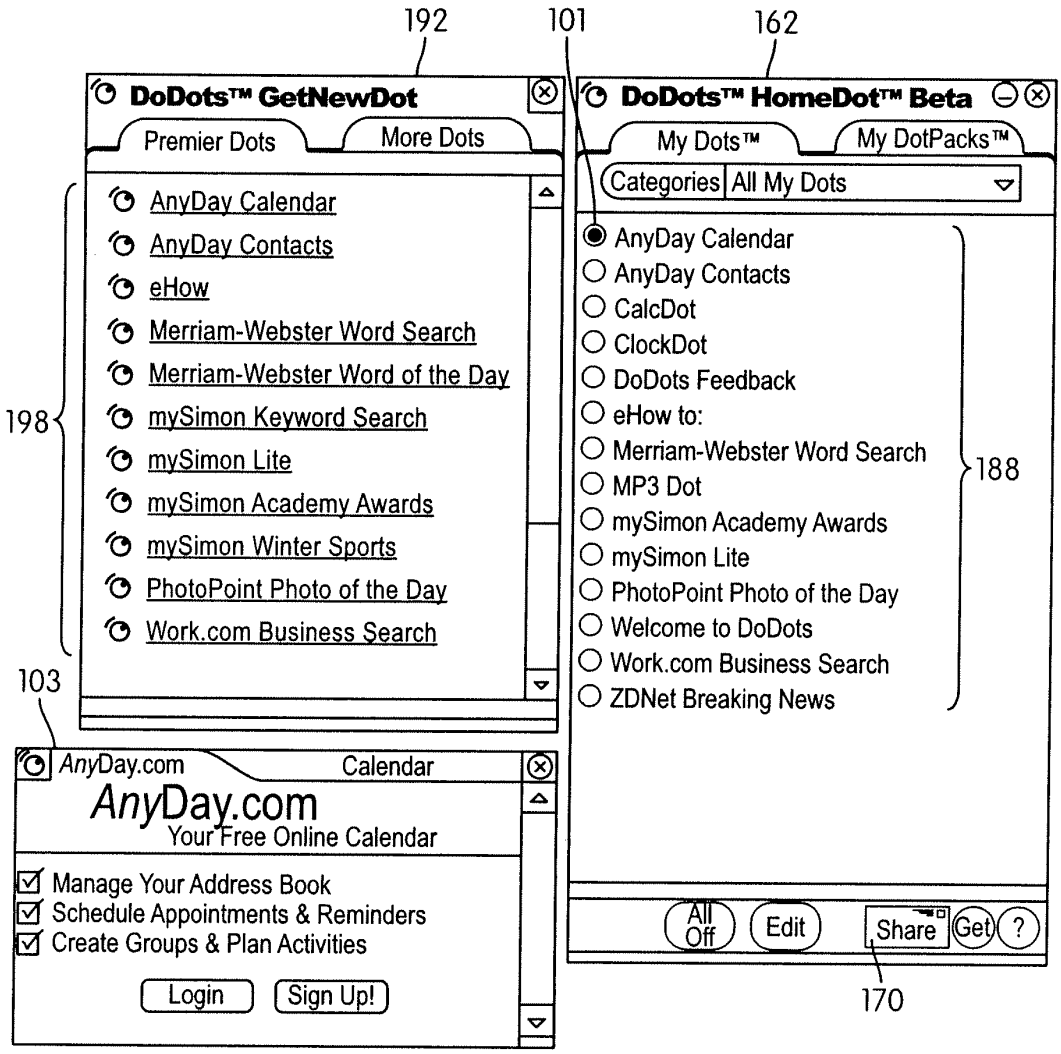

192
101
162
DoDots™ GetNewDot
Premier Dots
More Dots
AnyDay Calendar
AnyDay Contacts
eHow
Merriam-Webster Word Search
Merriam-Webster Word of the Day
mySimon Keyword Search
mySimon Lite
mySimon Academy Awards
mySimon Winter Sports
PhotoPoint Photo of the Day
Work.com Business Search
198
103
AnyDay.com
Calendar
AnyDay.com
Your Free Online Calendar
Manage Your Address Book
Schedule Appointments & Reminders
Create Groups & Plan Activities
Login
Sign Up!
DoDots™ HomeDot™ Beta
My Dots™
My DotPacks™
Categories
All My Dots
AnyDay Calendar
AnyDay Contacts
CalcDot
ClockDot
DoDots Feedback
eHow to:
Merriam-Webster Word Search
MP3 Dot
mySimon Academy Awards
mySimon Lite
PhotoPoint Photo of the Day
Welcome to DoDots
Work.com Business Search
ZDNet Breaking News
188
All Off
Edit
Share
Get
?
170

FIG. 6

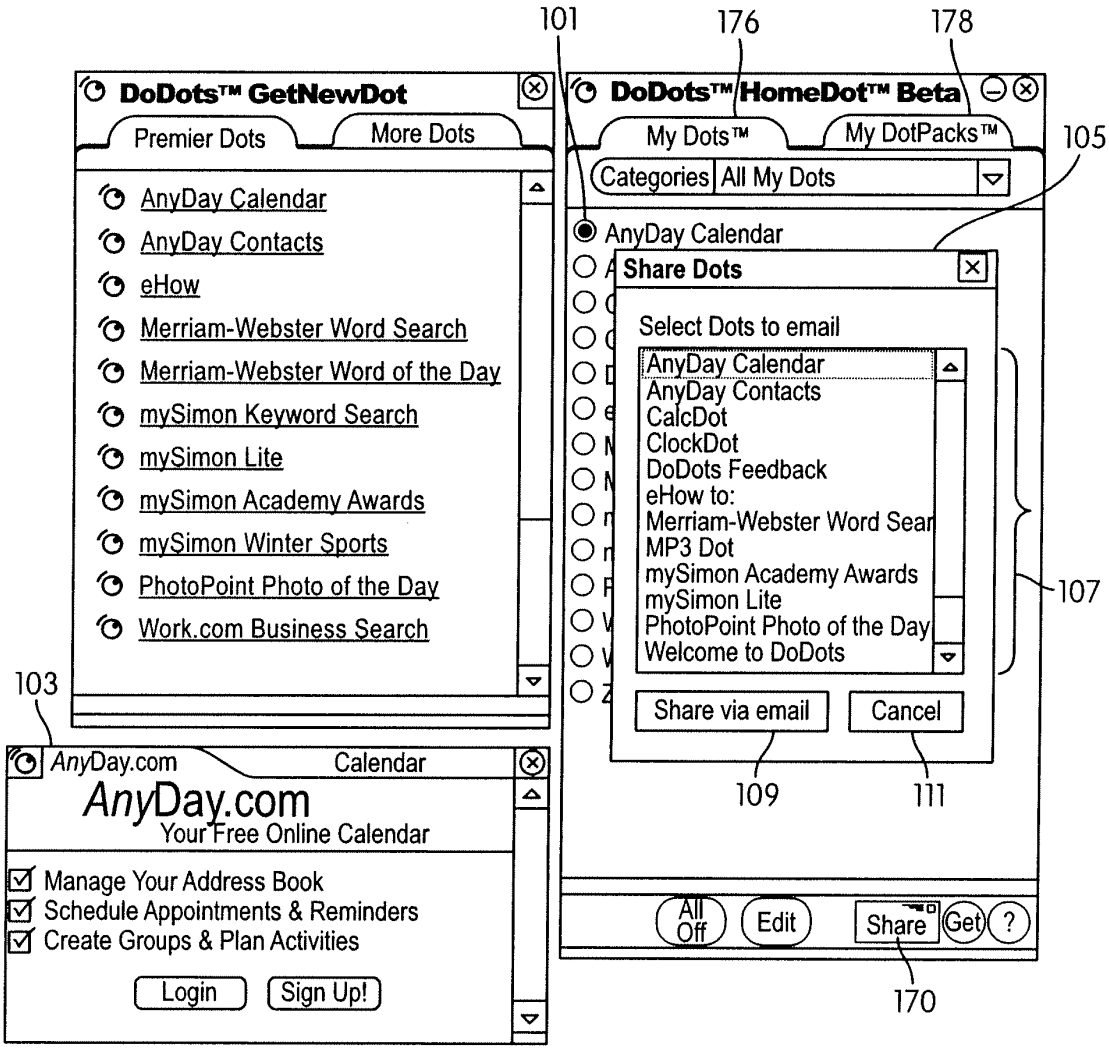

101
176
178
105
DoDots™ GetNewDot
Premier Dots
More Dots
AnyDay Calendar
AnyDay Contacts
eHow
Merriam-Webster Word Search
Merriam-Webster Word of the Day
mySimon Keyword Search
mySimon Lite
mySimon Academy Awards
mySimon Winter Sports
PhotoPoint Photo of the Day
Work.com Business Search
103
AnyDay.com
Calendar
AnyDay.com
Your Free Online Calendar
Manage Your Address Book
Schedule Appointments & Reminders
Create Groups & Plan Activities
Login
Sign Up!
DoDots™ HomeDot™ Beta
My Dots™
My DotPacks™
Categories
All My Dots
AnyDay Calendar
Share Dots
Select Dots to email
AnyDay Calendar
AnyDay Contacts
CalcDot
ClockDot
DoDots Feedback
eHow to:
Merriam-Webster Word Sear
MP3 Dot
mySimon Academy Awards
mySimon Lite
PhotoPoint Photo of the Day
Welcome to DoDots
107
Share via email
Cancel
109
111
All Off
Edit
Share
Get
?
170

FIG. 7

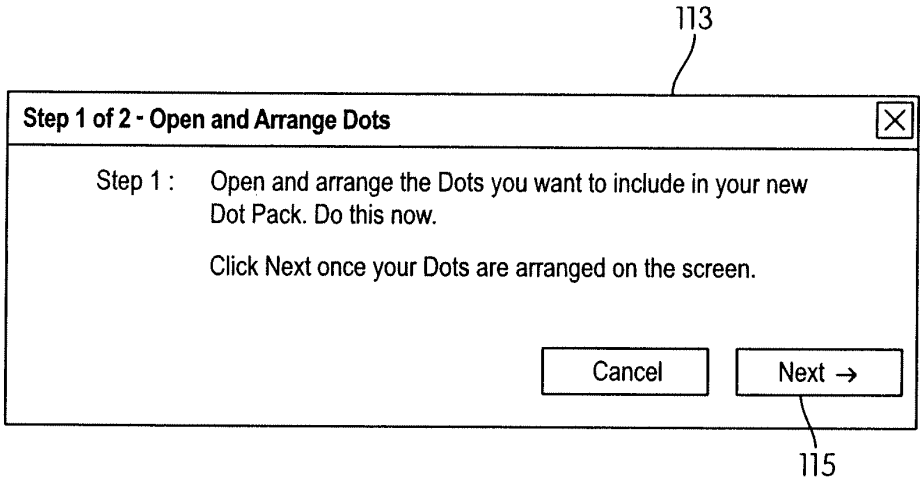

113
Step 1 of 2 - Open and Arrange Dots
Step 1 :
Open and arrange the Dots you want to include in your new Dot Pack. Do this now.
Click Next once your Dots are arranged on the screen.
Cancel
Next →
115

FIG. 8A

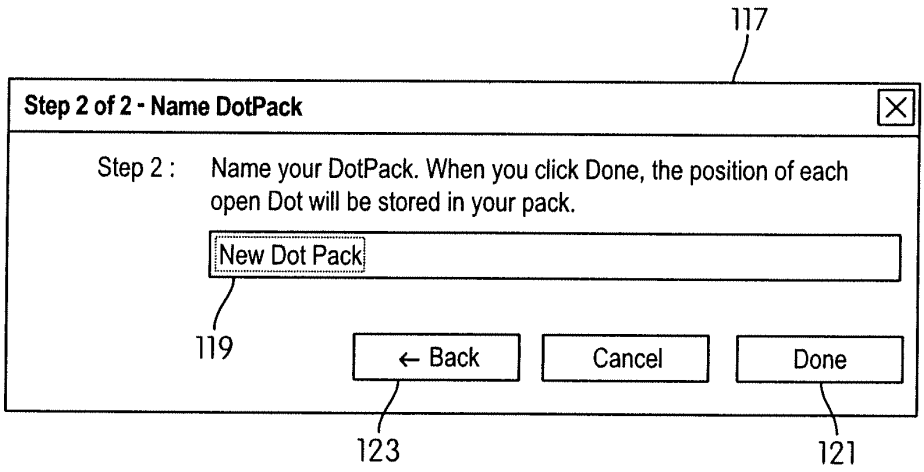

117
Step 2 of 2 - Name DotPack
Step 2 :
Name your DotPack. When you click Done, the position of each open Dot will be stored in your pack.
New Dot Pack
119
← Back
Cancel
Done
123
121

FIG. 8B

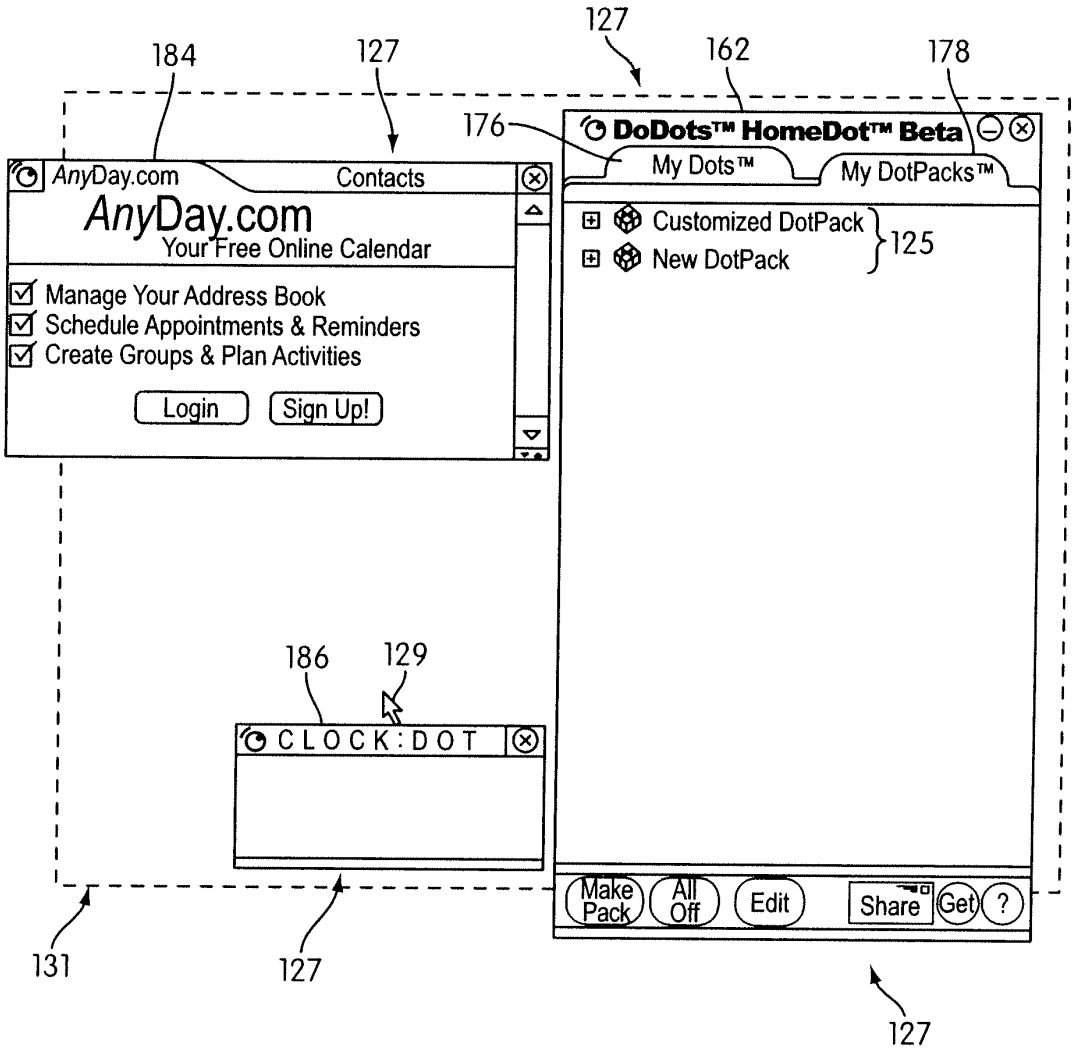

127
184
127
162
178
176
DoDots™ HomeDot™ Beta
AnyDay.com
Contacts
My Dots™
My DotPacks™
AnyDay.com
Your Free Online Calendar
Customized DotPack
New DotPack
125
Manage Your Address Book
Schedule Appointments & Reminders
Create Groups & Plan Activities
Login
Sign Up!
186
129
CLOCK:DOT
Make Pack
All Off
Edit
Share
Get
?
131
127
127

FIG. 9A

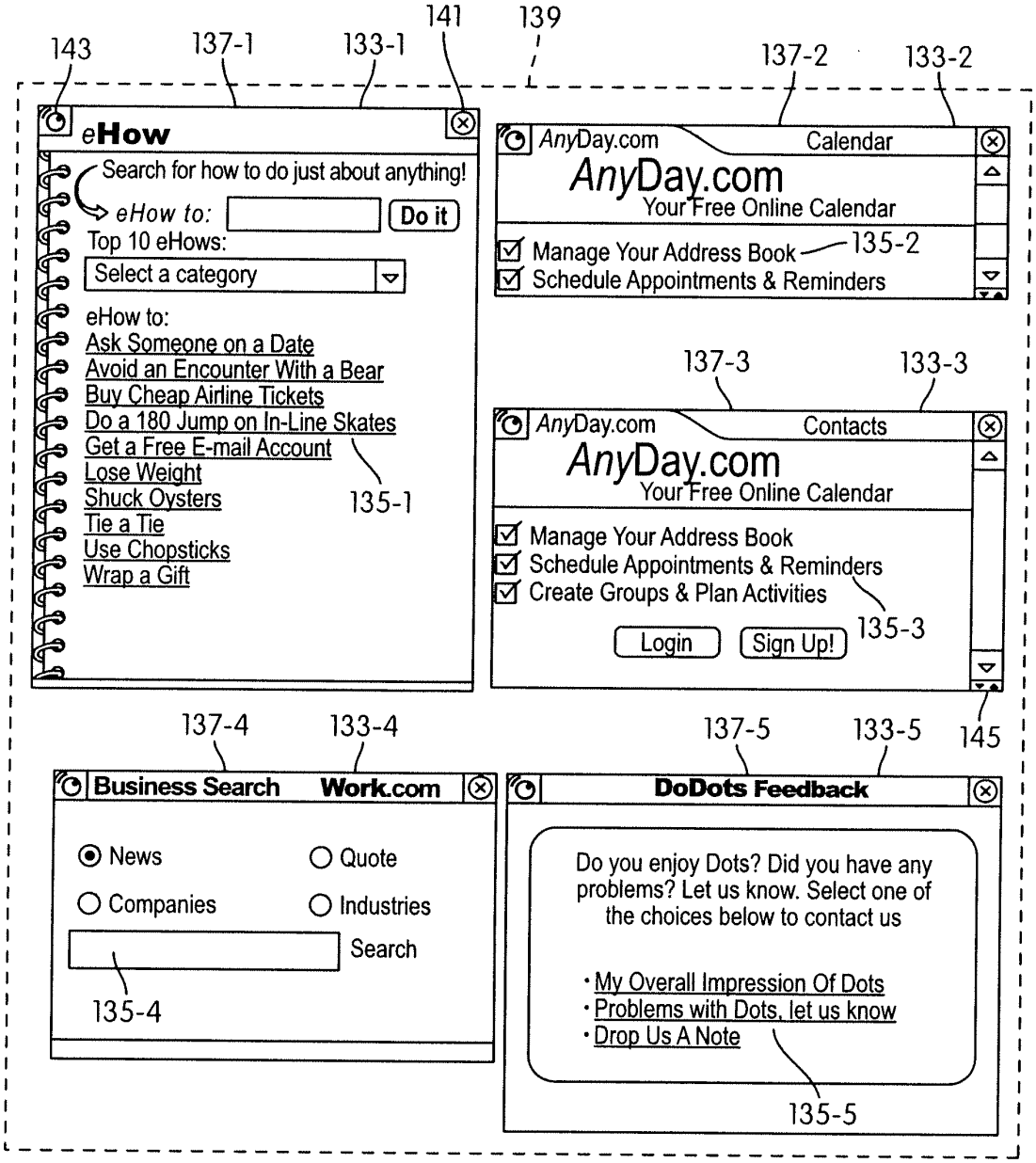

143
137-1
133-1
141
139
137-2
133-2
eHow
Search for how to do just about anything!
eHow to:
Do it
Top 10 eHows:
Select a category
eHow to:
Ask Someone on a Date
Avoid an Encounter With a Bear
Buy Cheap Airline Tickets
Do a 180 Jump on In-Line Skates
Get a Free E-mail Account
Lose Weight
Shuck Oysters
Tie a Tie
Use Chopsticks
Wrap a Gift
135-1
AnyDay.com
Calendar
AnyDay.com
Your Free Online Calendar
Manage Your Address Book
Schedule Appointments & Reminders
135-2
137-3
133-3
AnyDay.com
Contacts
AnyDay.com
Your Free Online Calendar
Manage Your Address Book
Schedule Appointments & Reminders
Create Groups & Plan Activities
Login
Sign Up!
135-3
145
137-4
133-4
Business Search
Work.com
News
Quote
Companies
Industries
Search
135-4
137-5
133-5
DoDots Feedback
Do you enjoy Dots? Did you have any problems? Let us know. Select one of the choices below to contact us
My Overall Impression Of Dots
Problems with Dots, let us know
Drop Us A Note
135-5

FIG. 9B

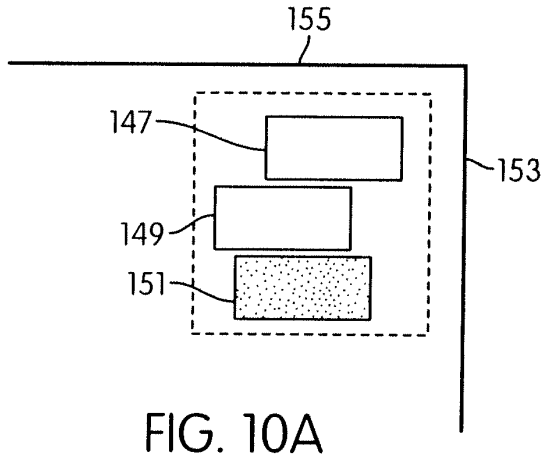


FIG. 10A

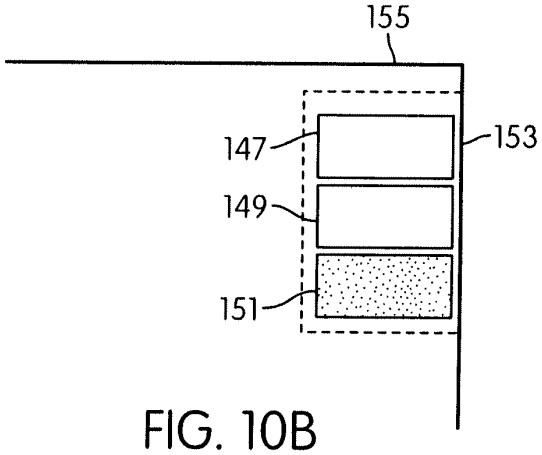


FIG. 10B

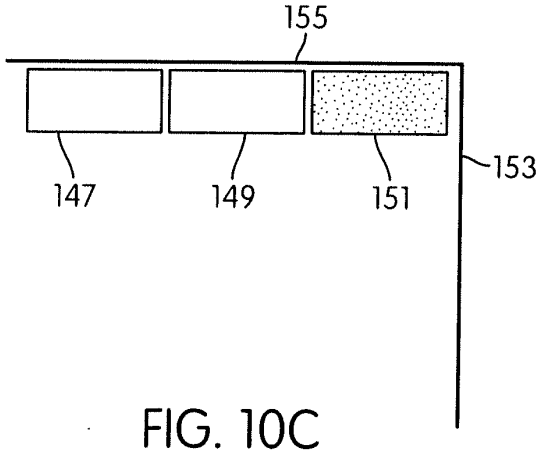


FIG. 10C

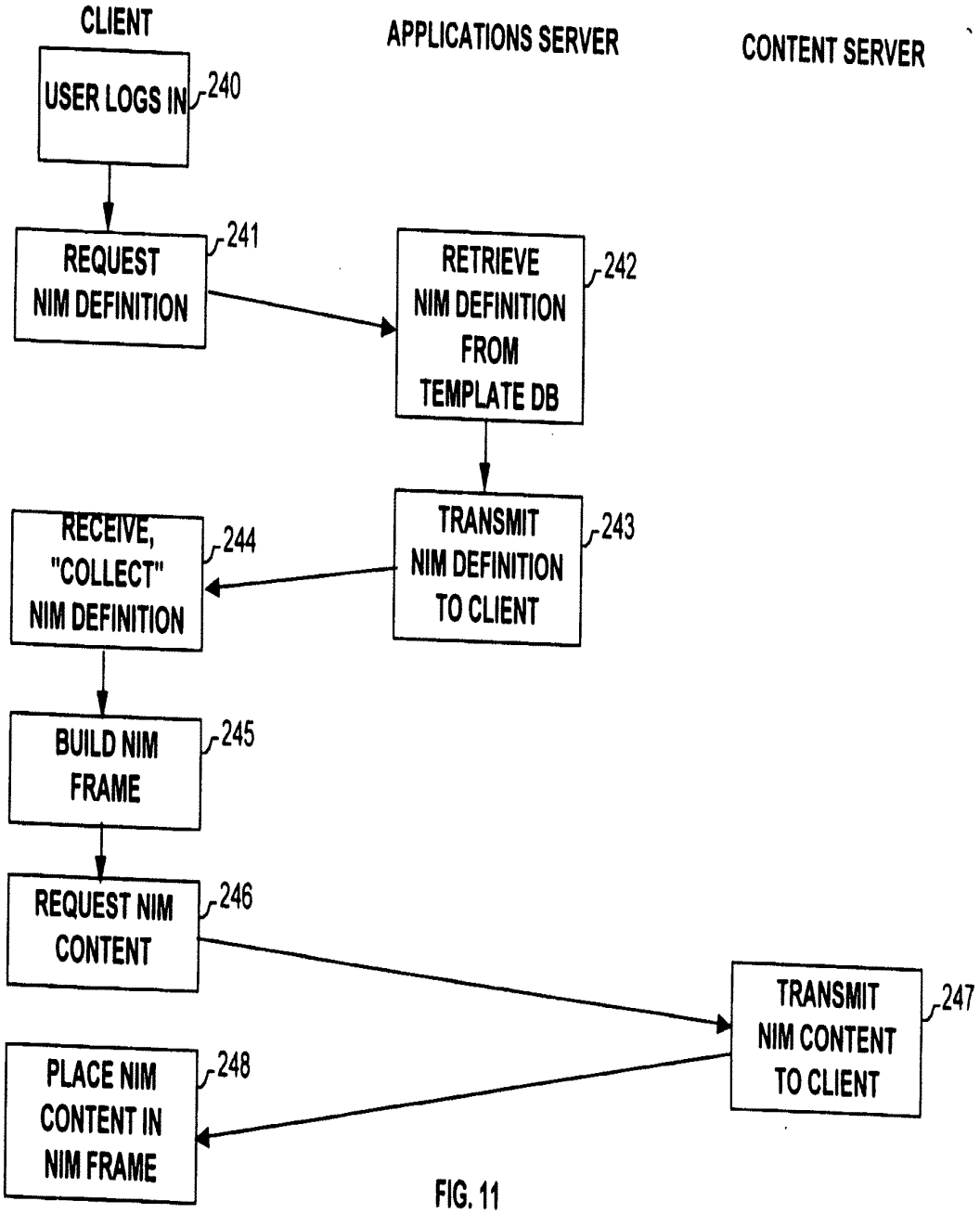

CLIENT
APPLICATIONS SERVER
CONTENT SERVER
USER LOGS IN
240
REQUEST NIM DEFINITION
241
RETRIEVE NIM DEFINITION FROM TEMPLATE DB
242
TRANSMIT NIM DEFINITION TO CLIENT
243
RECEIVE, "COLLECT" NIM DEFINITION
244
BUILD NIM FRAME
245
REQUEST NIM CONTENT
246
TRANSMIT NIM CONTENT TO CLIENT
247
PLACE NIM CONTENT IN NIM FRAME
248

FIG. 11

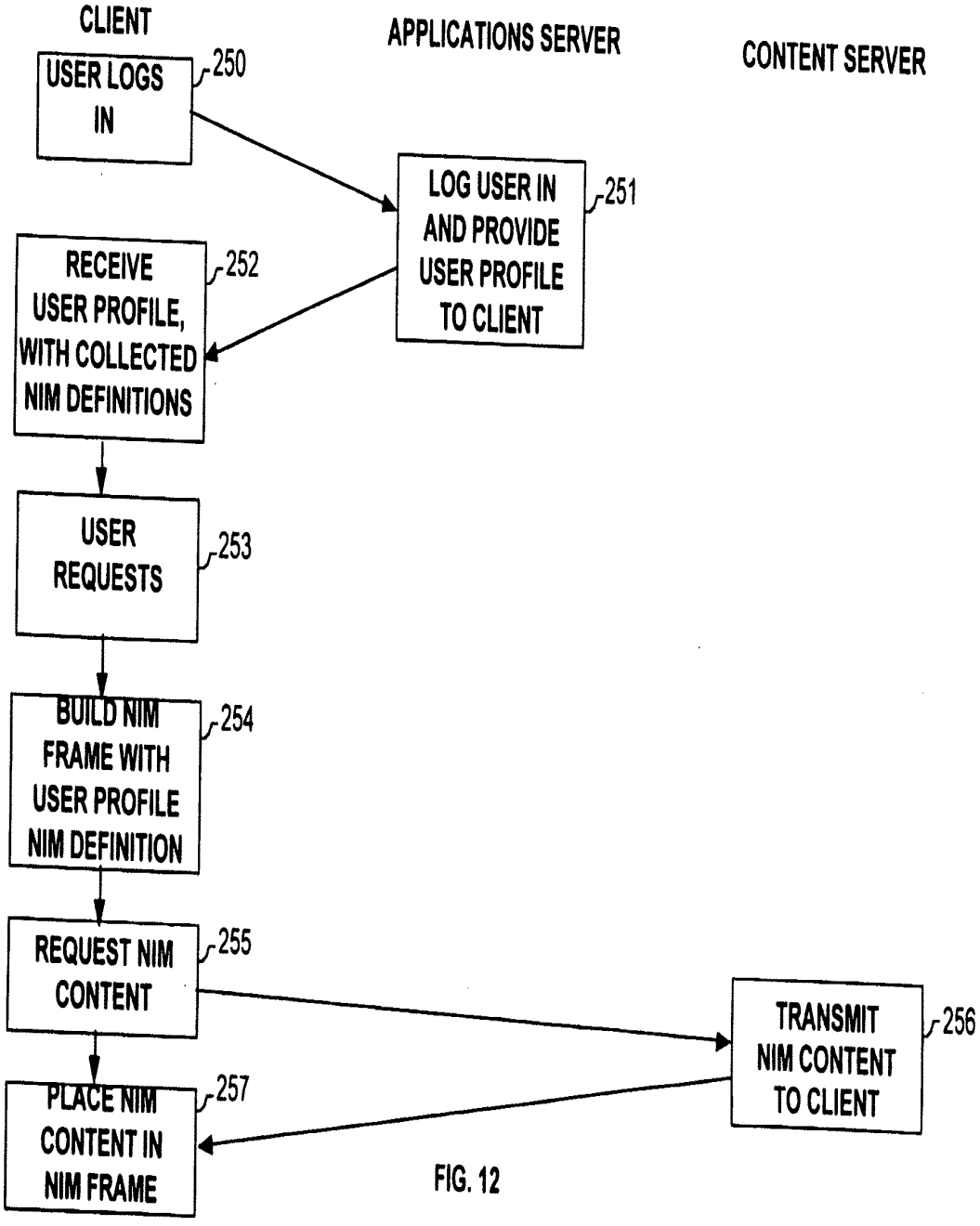

CLIENT
APPLICATIONS SERVER
CONTENT SERVER
USER LOGS IN
250
LOG USER IN AND PROVIDE USER PROFILE TO CLIENT
251
RECEIVE USER PROFILE, WITH COLLECTED NIM DEFINITIONS
252
USER REQUESTS
253
BUILD NIM FRAME WITH USER PROFILE NIM DEFINITION
254
REQUEST NIM CONTENT
255
TRANSMIT NIM CONTENT TO CLIENT
256
PLACE NIM CONTENT IN NIM FRAME
257

FIG. 12

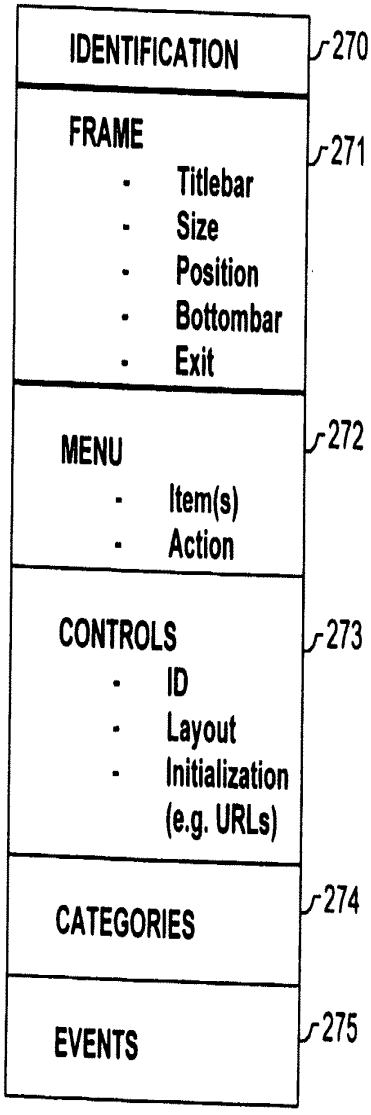

IDENTIFICATION
270
FRAME
- Titlebar
- Size
- Position
- Bottombar
- Exit
271
MENU
- Item(s)
- Action
272
CONTROLS
- ID
- Layout
- Initialization (e.g. URLs)
273
CATEGORIES
274
EVENTS
275

FIG. 13

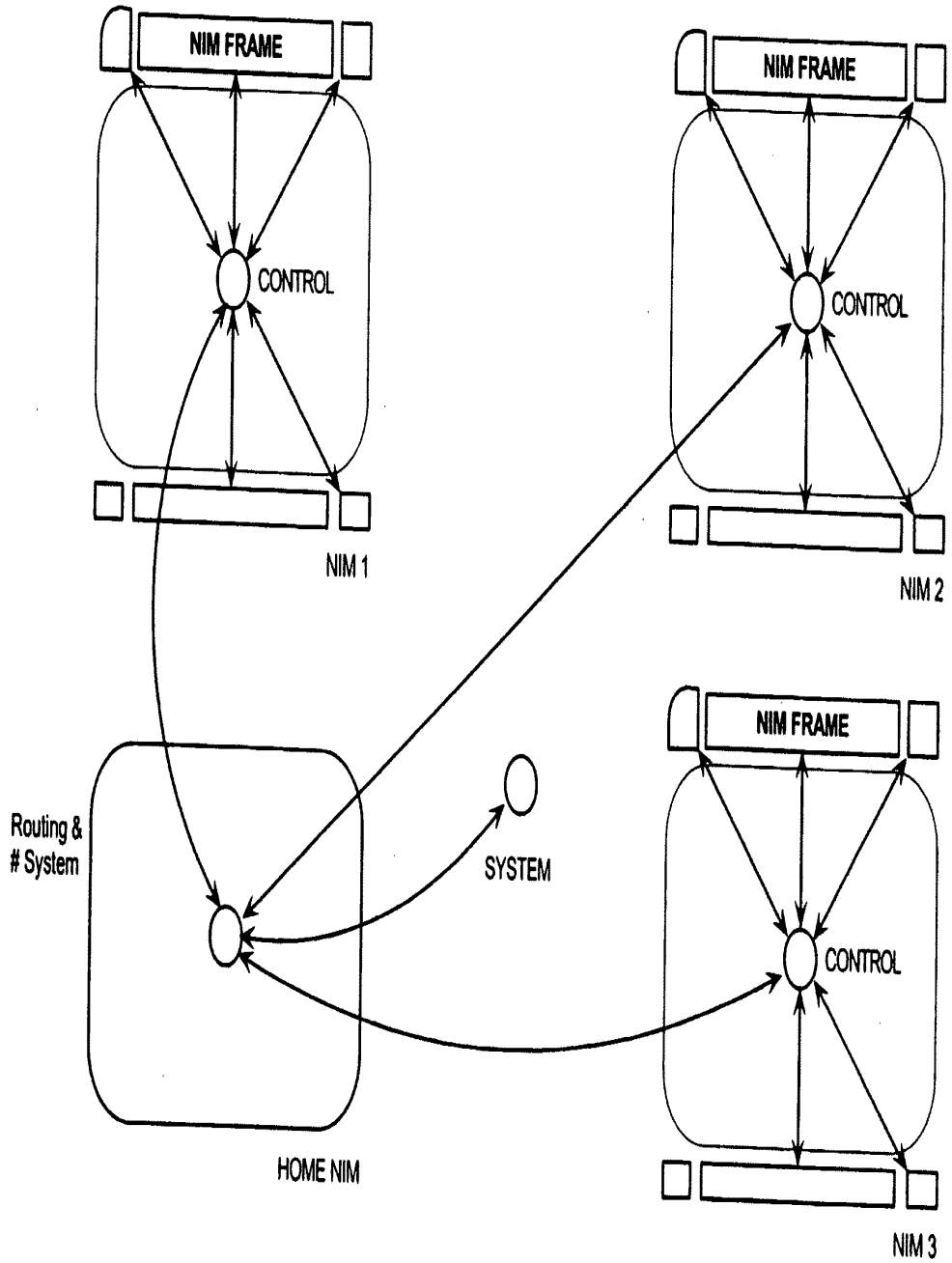

NIM FRAME
CONTROL
NIM 1
NIM FRAME
CONTROL
NIM 2
NIM FRAME
CONTROL
NIM 3
Routing &
System
SYSTEM
HOME NIM

FIG. 14

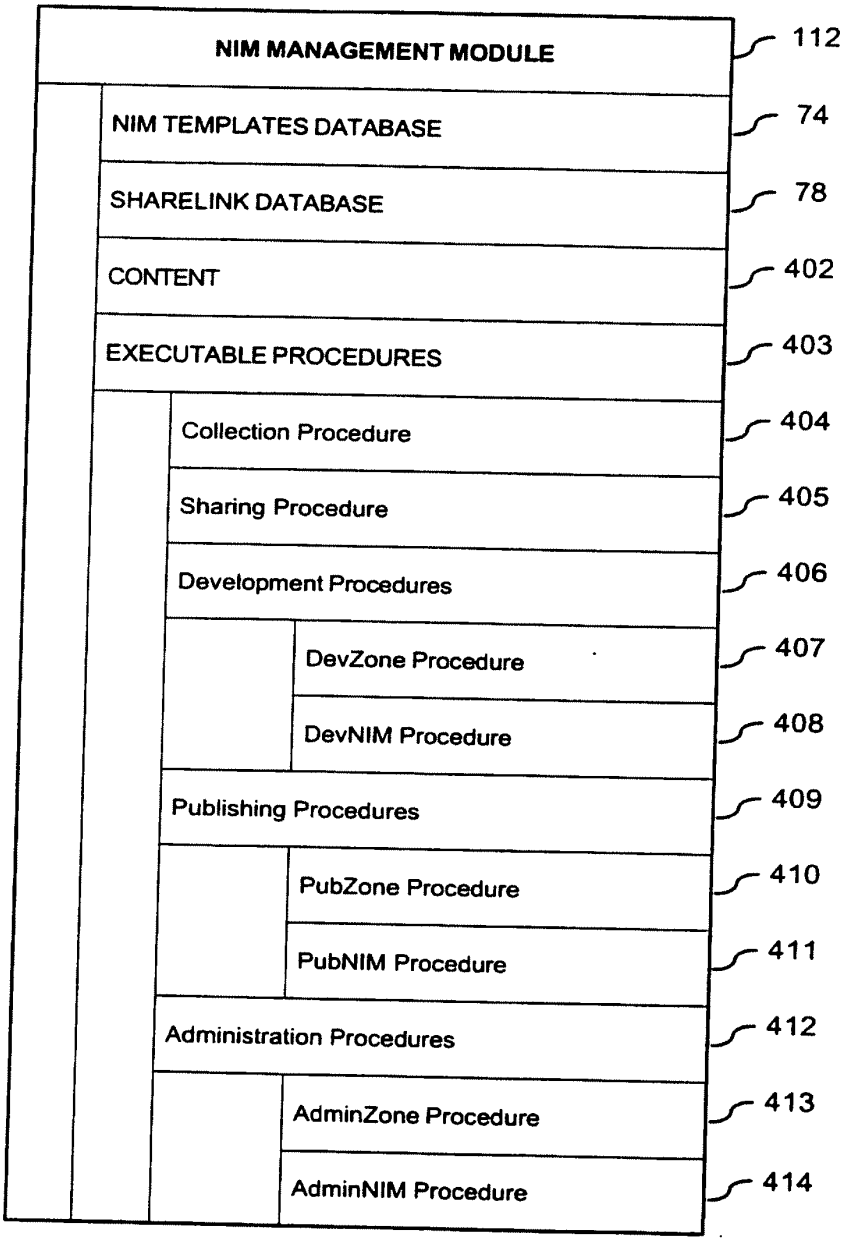

NIM MANAGEMENT MODULE
112
NIM TEMPLATES DATABASE
74
SHARELINK DATABASE
78
CONTENT
402
EXECUTABLE PROCEDURES
403
Collection Procedure
404
Sharing Procedure
405
Development Procedures
406
DevZone Procedure
407
DevNIM Procedure
408
Publishing Procedures
409
PubZone Procedure
410
PubNIM Procedure
411
Administration Procedures
412
AdminZone Procedure
413
AdminNIM Procedure
414

FIG. 15

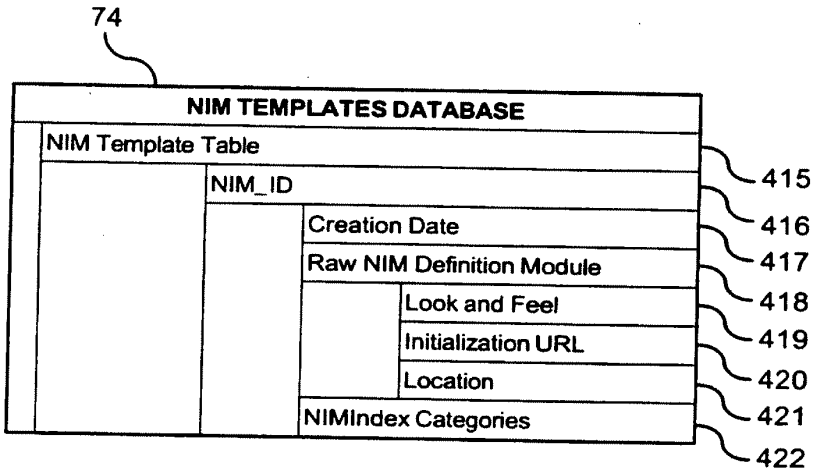

74
NIM TEMPLATES DATABASE
NIM Template Table
NIM_ID
Creation Date
Raw NIM Definition Module
Look and Feel
Initialization URL
Location
NIMIndex Categories
415
416
417
418
419
420
421
422

FIG. 16

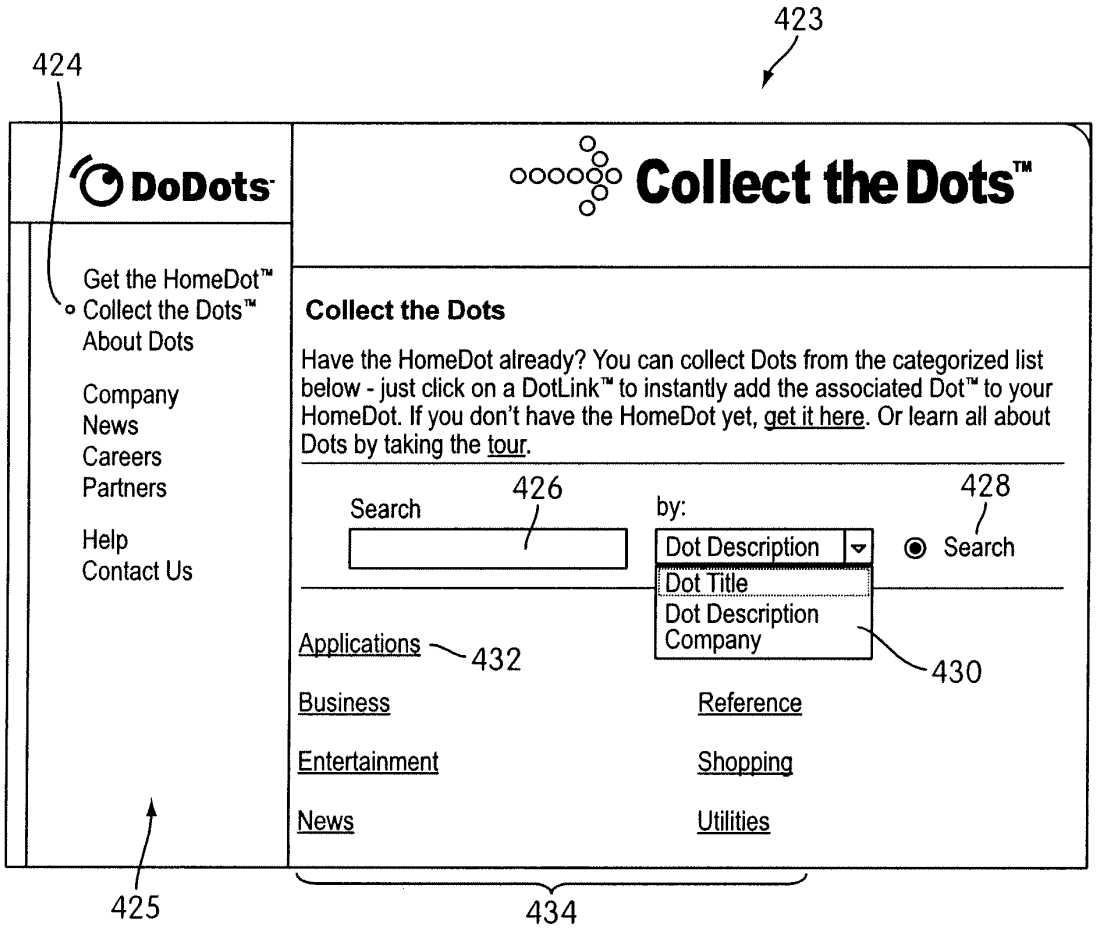

423
424
DoDots
Collect the Dots™
Get the HomeDot™
Collect the Dots™
About Dots
Company
News
Careers
Partners
Help
Contact Us
Collect the Dots
Have the HomeDot already? You can collect Dots from the categorized list below - just click on a DotLink™ to instantly add the associated Dot™ to your HomeDot. If you don't have the HomeDot yet, get it here. Or learn all about Dots by taking the tour.
Search
426
by:
428
Dot Description
Search
Dot Title
Dot Description
Company
430
Applications
432
Business
Reference
Entertainment
Shopping
News
Utilities
425
434

FIG. 17

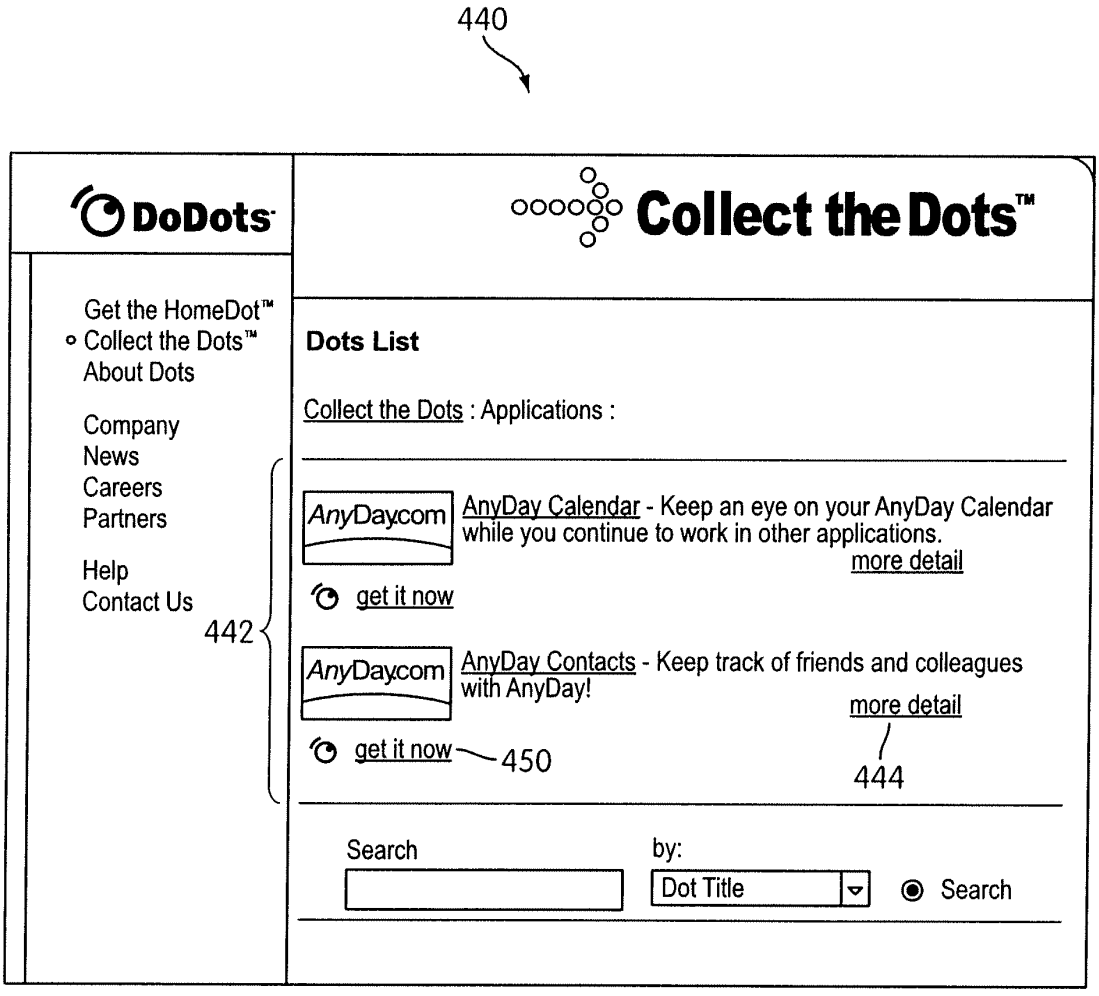

440
DoDots
Collect the Dots™
Get the HomeDot™
Collect the Dots™
About Dots
Company
News
Careers
Partners
Help
Contact Us
442
Dots List
Collect the Dots : Applications :
AnyDay.com
AnyDay Calendar - Keep an eye on your AnyDay Calendar while you continue to work in other applications.
more detail
get it now
AnyDay.com
AnyDay Contacts - Keep track of friends and colleagues with AnyDay!
more detail
get it now
450
444
Search
by:
Dot Title
Search

FIG. 18

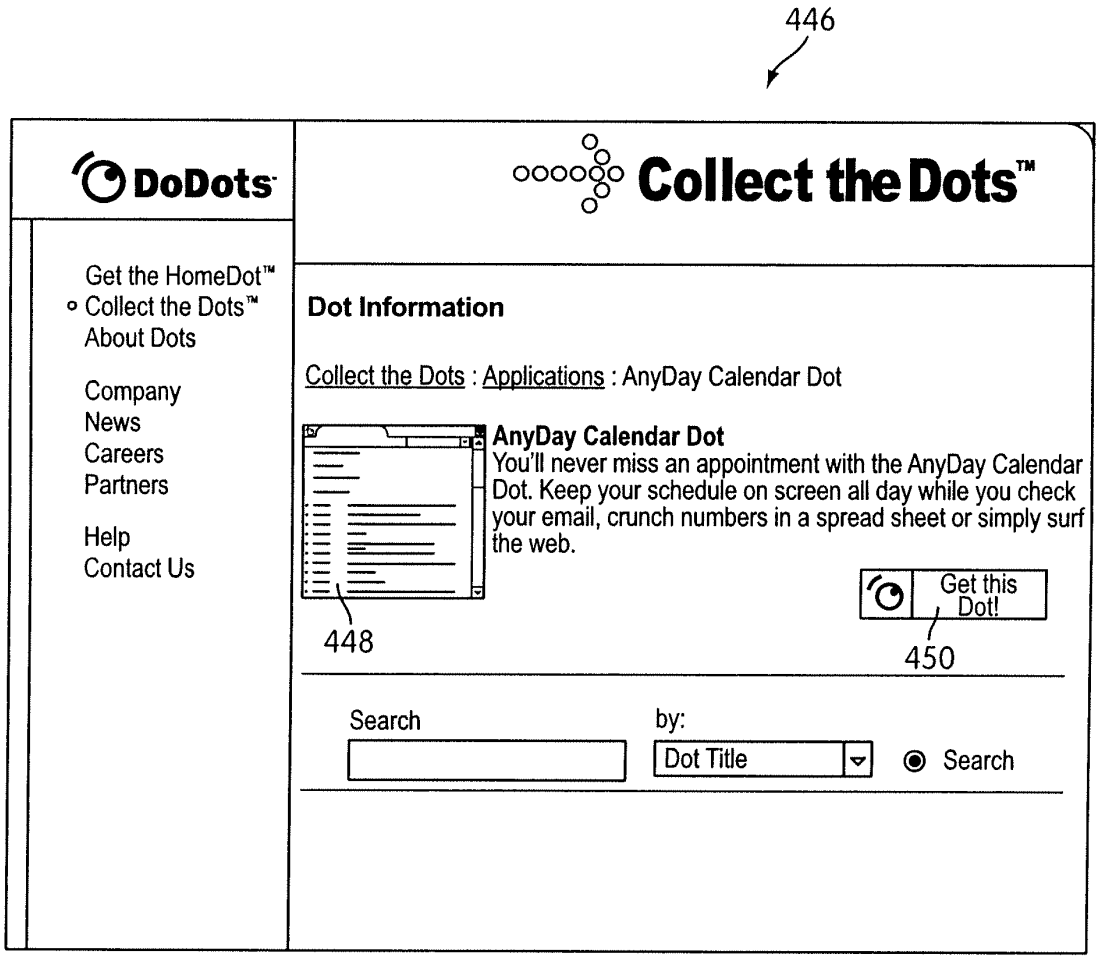

446
DoDots
Collect the Dots™
Get the HomeDot™
Collect the Dots™
About Dots
Company
News
Careers
Partners
Help
Contact Us
Dot Information
Collect the Dots : Applications : AnyDay Calendar Dot
AnyDay Calendar Dot
You'll never miss an appointment with the AnyDay Calendar Dot. Keep your schedule on screen all day while you check your email, crunch numbers in a spread sheet or simply surf the web.
Get this Dot!
448
450
Search
by:
Dot Title
Search

FIG. 19

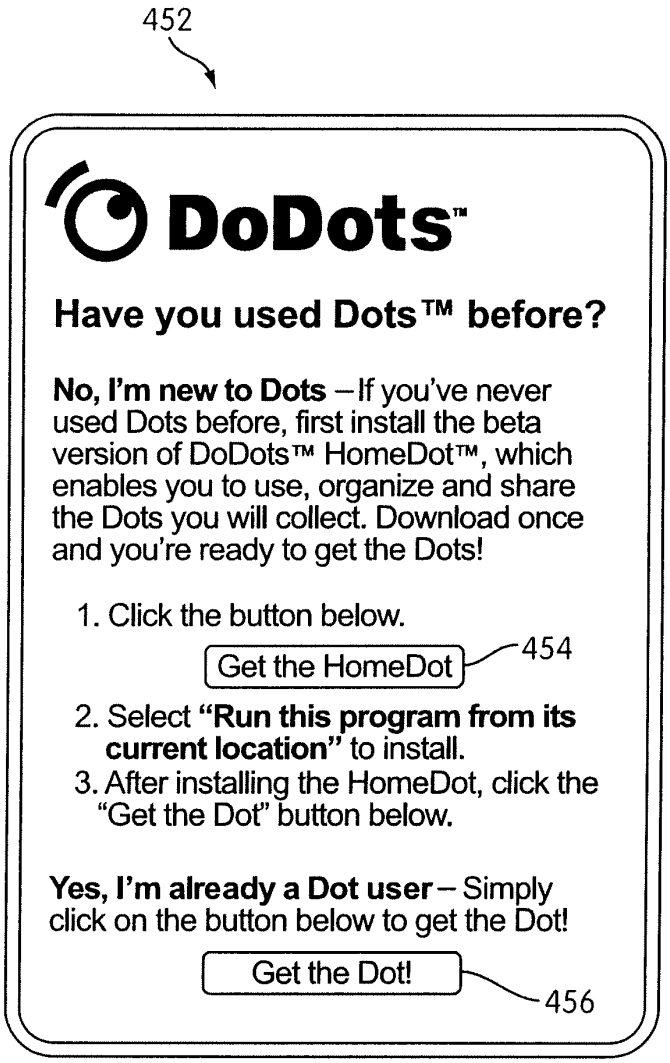

452
DoDots™
Have you used Dots™ before?
No, I'm new to Dots – If you've never used Dots before, first install the beta version of DoDots™ HomeDot™, which enables you to use, organize and share the Dots you will collect. Download once and you're ready to get the Dots!
1. Click the button below.
Get the HomeDot
454
2. Select "Run this program from its current location" to install.
3. After installing the HomeDot, click the "Get the Dot" button below.
Yes, I'm already a Dot user – Simply click on the button below to get the Dot!
Get the Dot!
456

FIG. 20

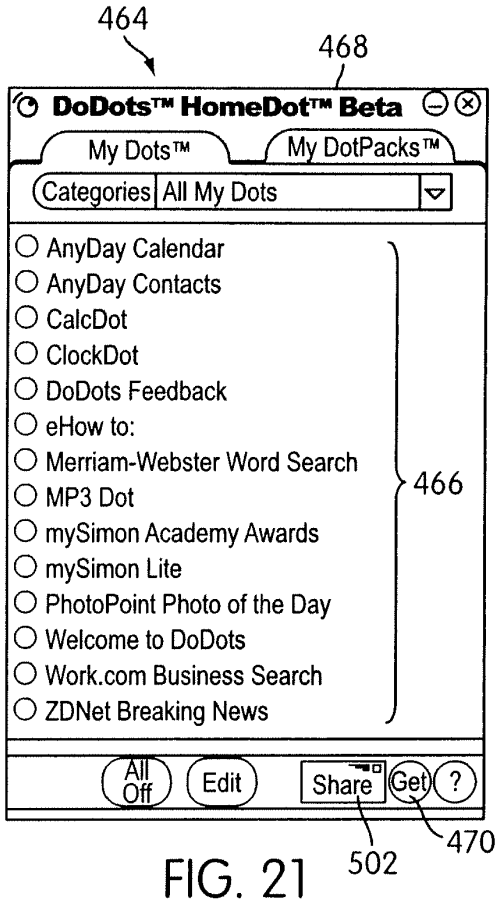


FIG. 21

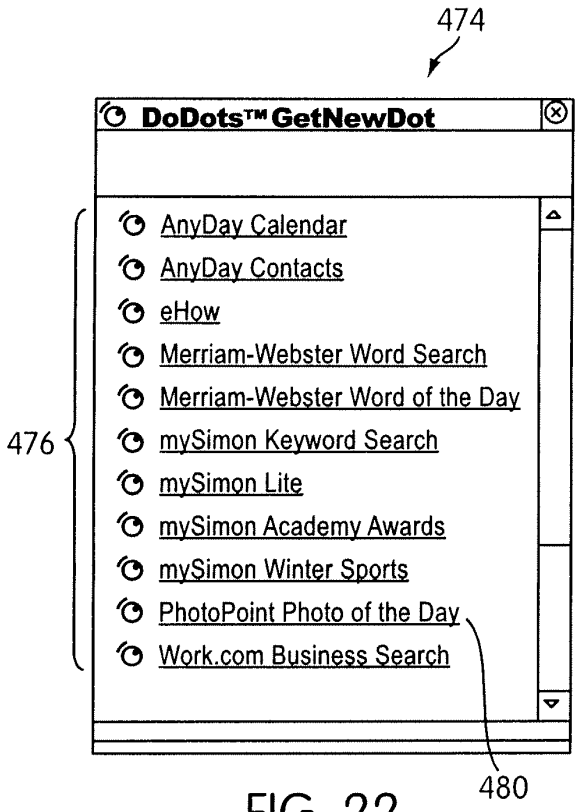


FIG. 22

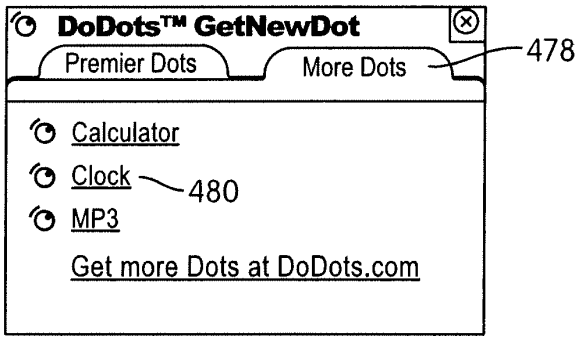


FIG. 23

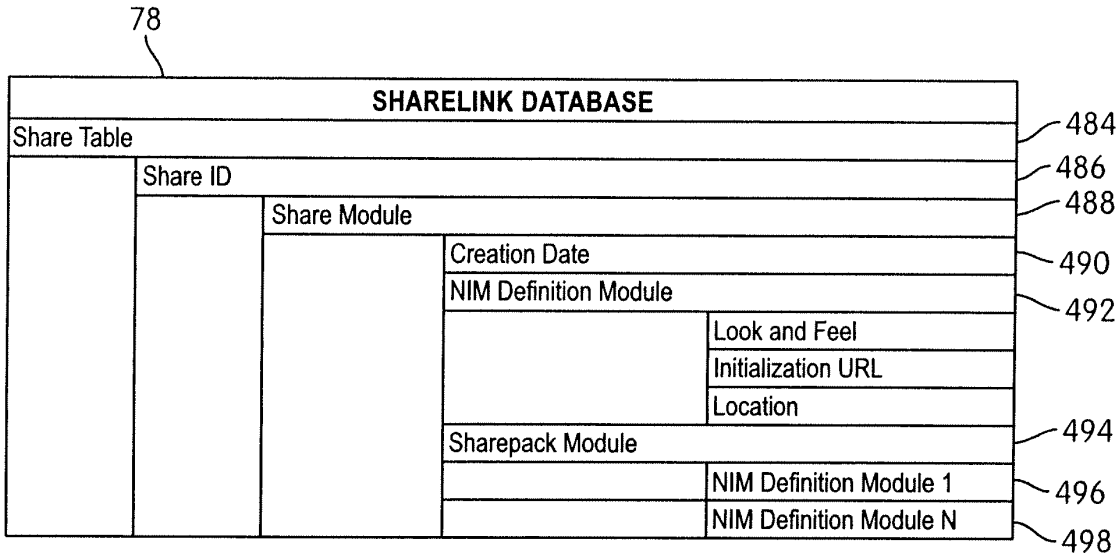

78
SHARELINK DATABASE
Share Table
Share ID
Share Module
Creation Date
NIM Definition Module
Look and Feel
Initialization URL
Location
Sharepack Module
NIM Definition Module 1
NIM Definition Module N
484
486
488
490
492
494
496
498

FIG. 24

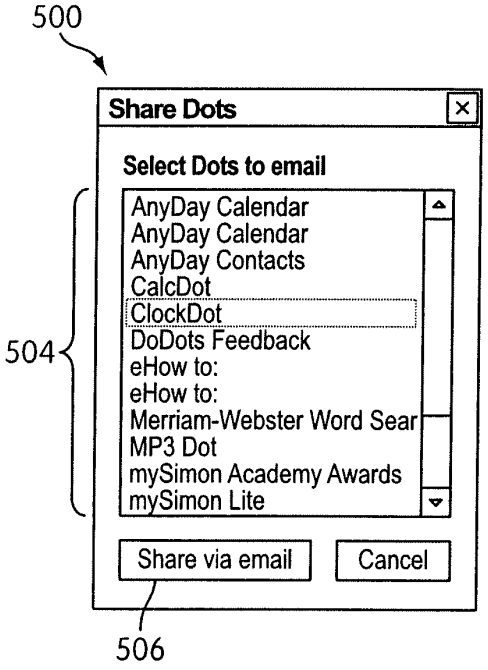

500
Share Dots
Select Dots to email
AnyDay Calendar
AnyDay Calendar
AnyDay Contacts
CalcDot
ClockDot
DoDots Feedback
eHow to:
eHow to:
Merriam-Webster Word Sear
MP3 Dot
mySimon Academy Awards
mySimon Lite
504
Share via email
Cancel
506

FIG. 25

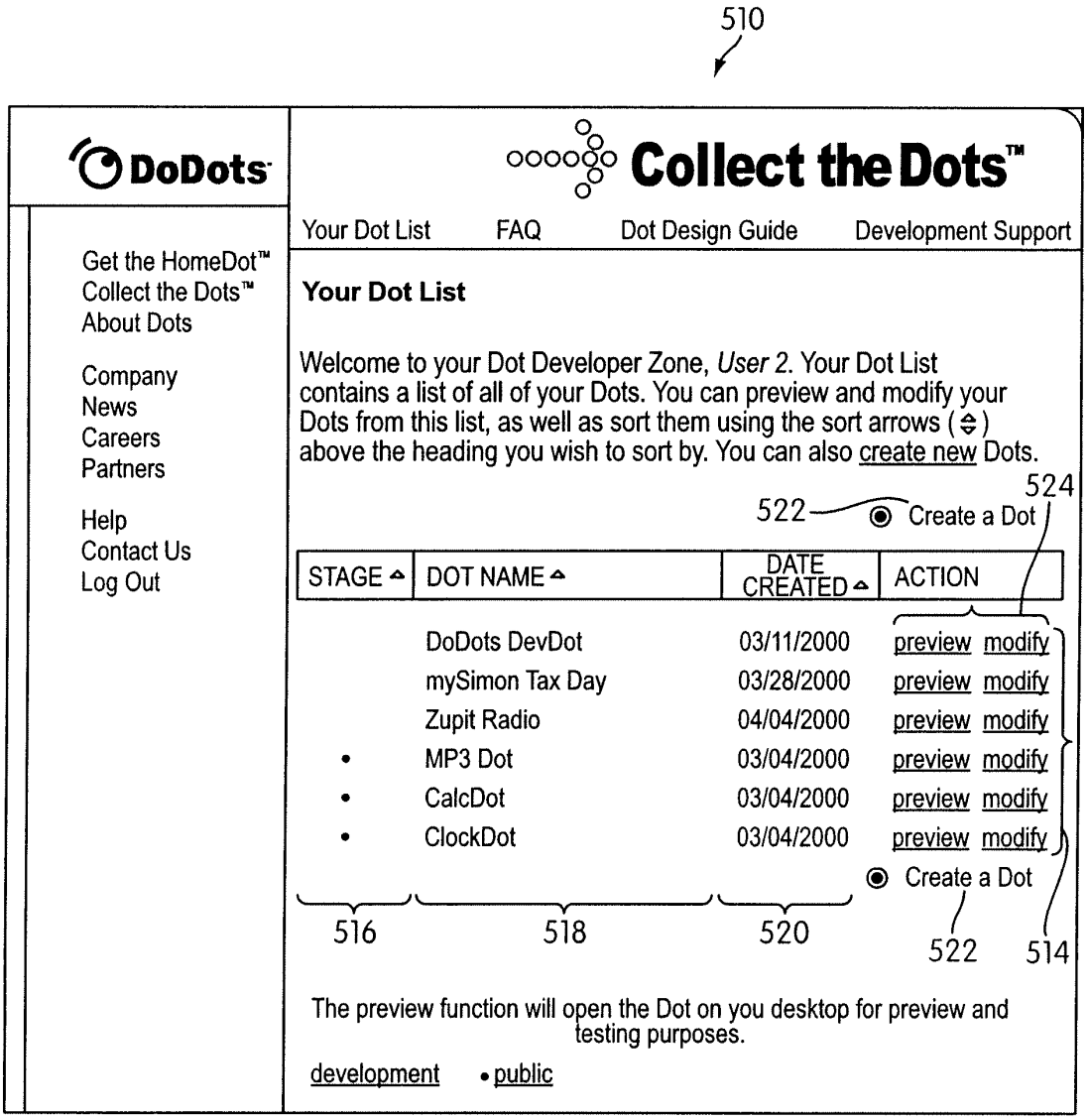

510
DoDots
Collect the Dots™
Your Dot List
FAQ
Dot Design Guide
Development Support
Get the HomeDot™
Collect the Dots™
About Dots
Company
News
Careers
Partners
Help
Contact Us
Log Out
Your Dot List
Welcome to your Dot Developer Zone, User 2. Your Dot List contains a list of all of your Dots. You can preview and modify your Dots from this list, as well as sort them using the sort arrows (⇳) above the heading you wish to sort by. You can also create new Dots.
522
Create a Dot
524
STAGE
DOT NAME
DATE CREATED
ACTION
DoDots DevDot 03/11/2000 preview modify
mySimon Tax Day 03/28/2000 preview modify
Zupit Radio 04/04/2000 preview modify
MP3 Dot 03/04/2000 preview modify
CalcDot 03/04/2000 preview modify
ClockDot 03/04/2000 preview modify
Create a Dot
516
518
520
522
514
The preview function will open the Dot on you desktop for preview and testing purposes.
development
public

FIG. 26

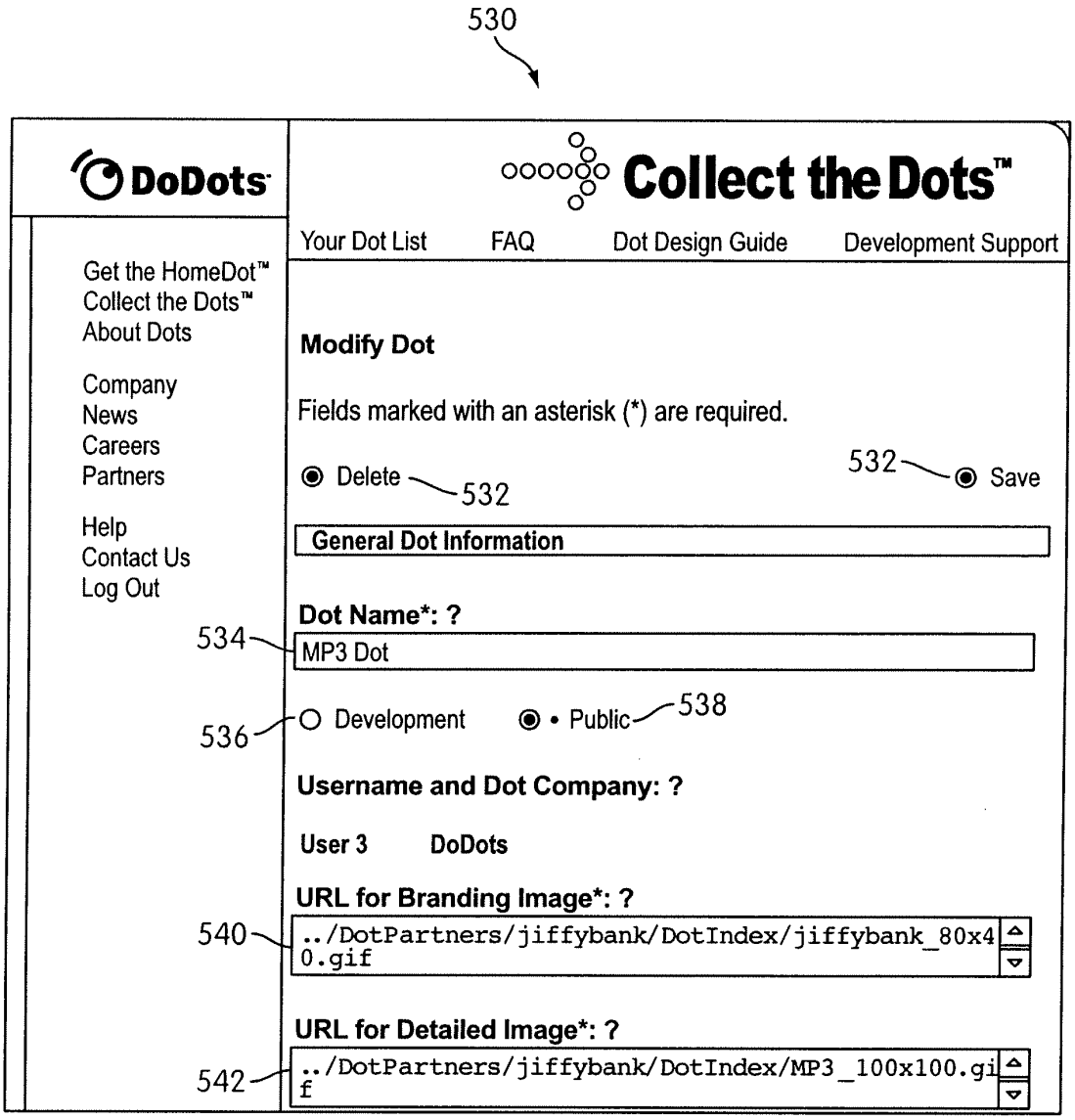

530
DoDots
Collect the Dots™
Your Dot List
FAQ
Dot Design Guide
Development Support
Get the HomeDot™
Collect the Dots™
About Dots
Company
News
Careers
Partners
Help
Contact Us
Log Out
Modify Dot
Fields marked with an asterisk (*) are required.
Delete
532
532
Save
General Dot Information
Dot Name*: ?
534
MP3 Dot
536
Development
Public
538
Username and Dot Company: ?
User 3
DoDots
URL for Branding Image*: ?
540
../DotPartners/jiffybank/DotIndex/jiffybank_80x40.gif
URL for Detailed Image*: ?
542
../DotPartners/jiffybank/DotIndex/MP3_100x100.gif

FIG. 27

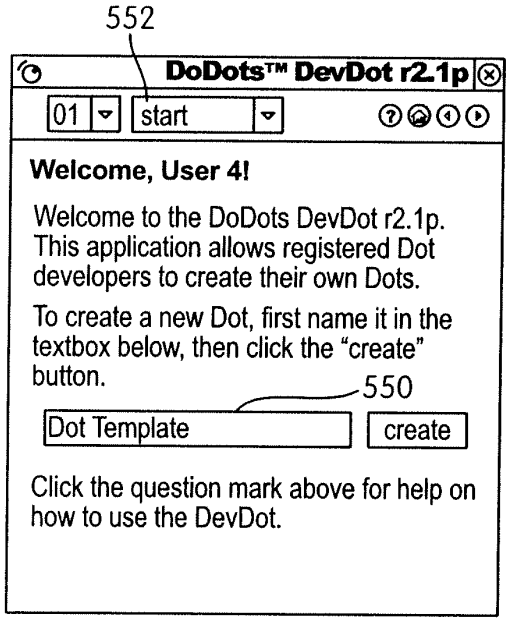


FIG. 28A

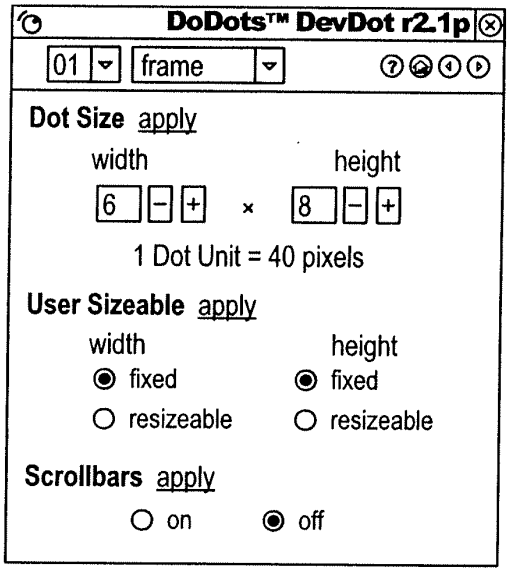


FIG. 28B

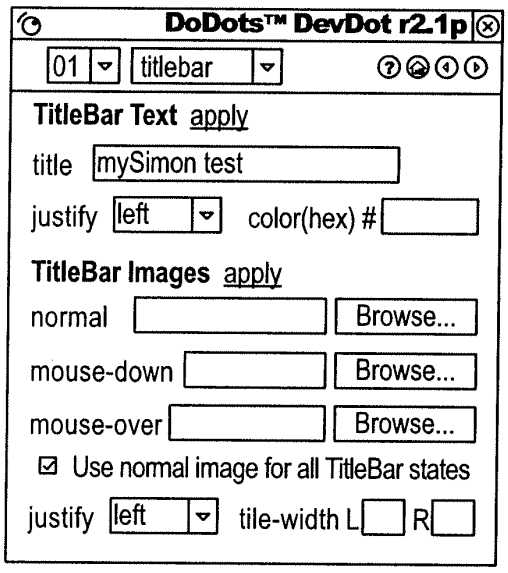


FIG. 28C

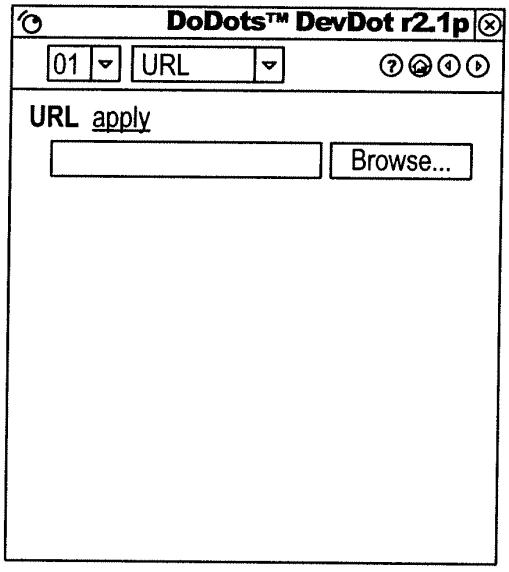


FIG. 28D

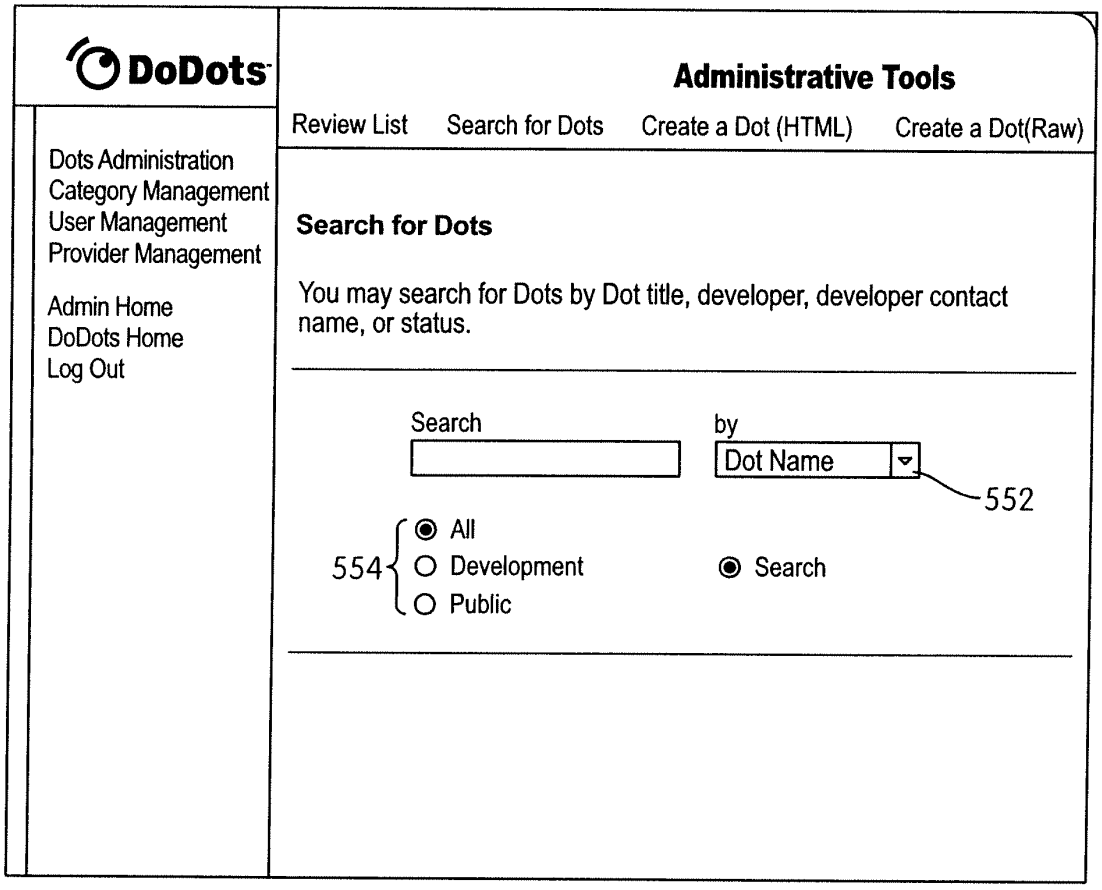

DoDots
Administrative Tools
Review List
Search for Dots
Create a Dot (HTML)
Create a Dot(Raw)
Dots Administration
Category Management
User Management
Provider Management
Admin Home
DoDots Home
Log Out
Search for Dots
You may search for Dots by Dot title, developer, developer contact name, or status.
Search
by
Dot Name
552
554
All
Development
Public
Search

FIG. 29A

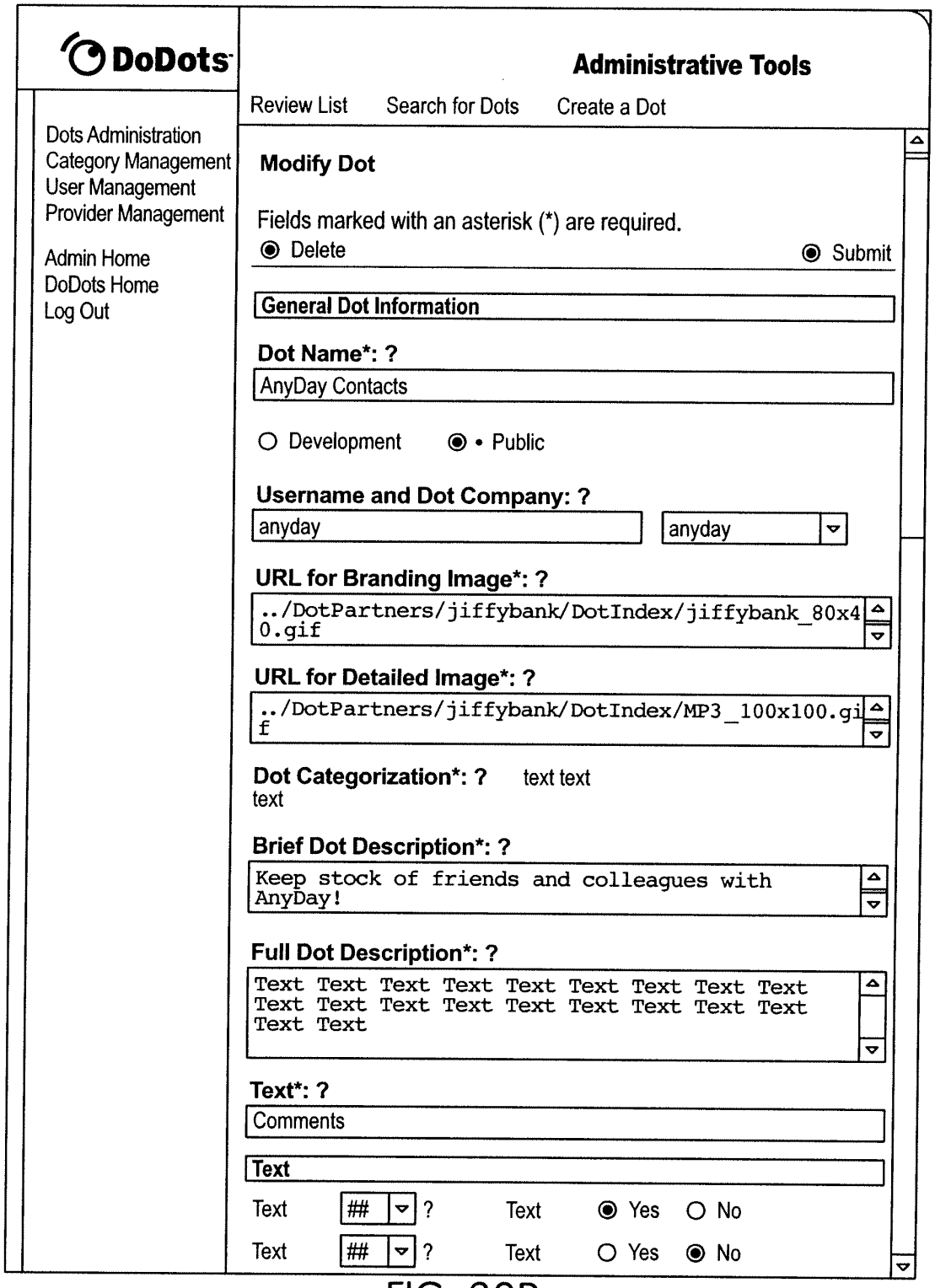

DoDots
Administrative Tools
Review List
Search for Dots
Create a Dot
Dots Administration
Category Management
User Management
Provider Management
Admin Home
DoDots Home
Log Out
Modify Dot
Fields marked with an asterisk (*) are required.
Delete
Submit
General Dot Information
Dot Name*: ?
AnyDay Contacts
Development
Public
Username and Dot Company: ?
anyday
anyday
URL for Branding Image*: ?
../DotPartners/jiffybank/DotIndex/jiffybank_80x40.gif
URL for Detailed Image*: ?
../DotPartners/jiffybank/DotIndex/MP3_100x100.gif
Dot Categorization*: ?
text text
text
Brief Dot Description*: ?
Keep stock of friends and colleagues with AnyDay!
Full Dot Description*: ?
Text Text Text Text Text Text Text Text Text
Text Text Text Text Text Text Text Text Text
Text Text
Text*: ?
Comments
Text
Text
##
?
Text
Yes
No
Text
##
?
Text
Yes
No

FIG. 29B

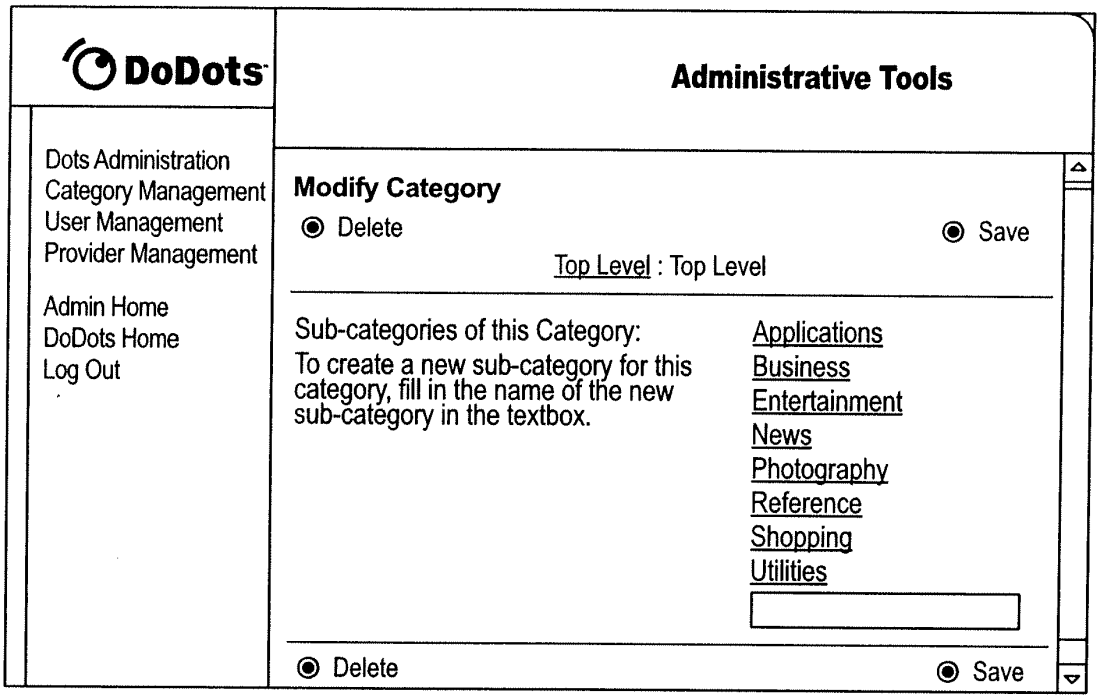


FIG. 30A

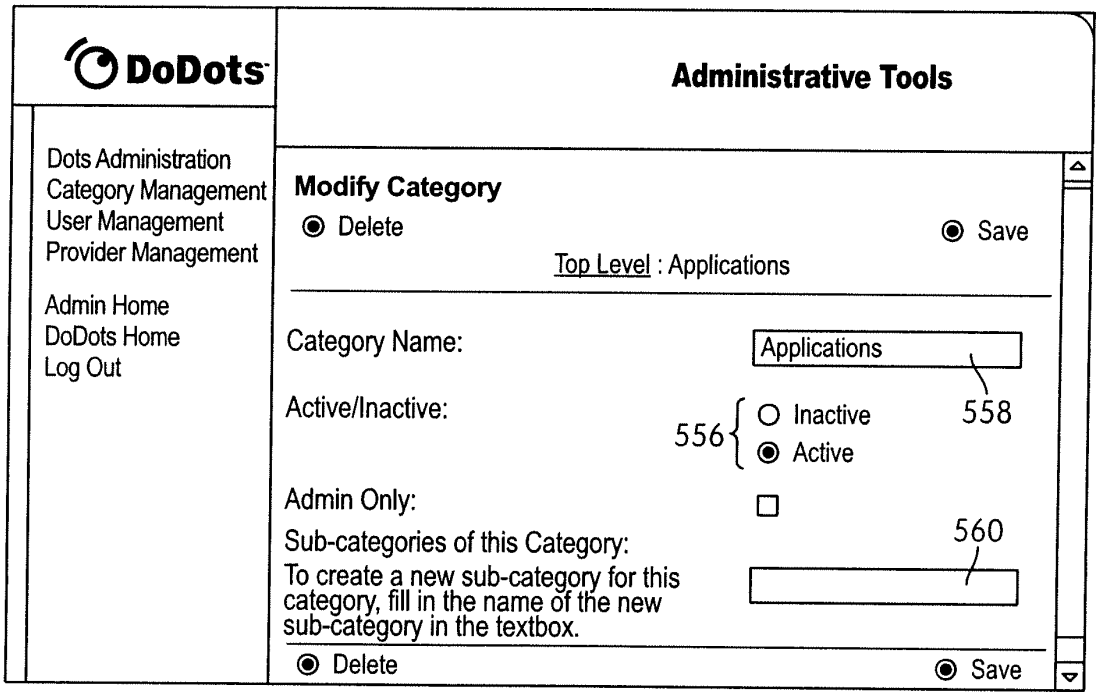


FIG. 30B

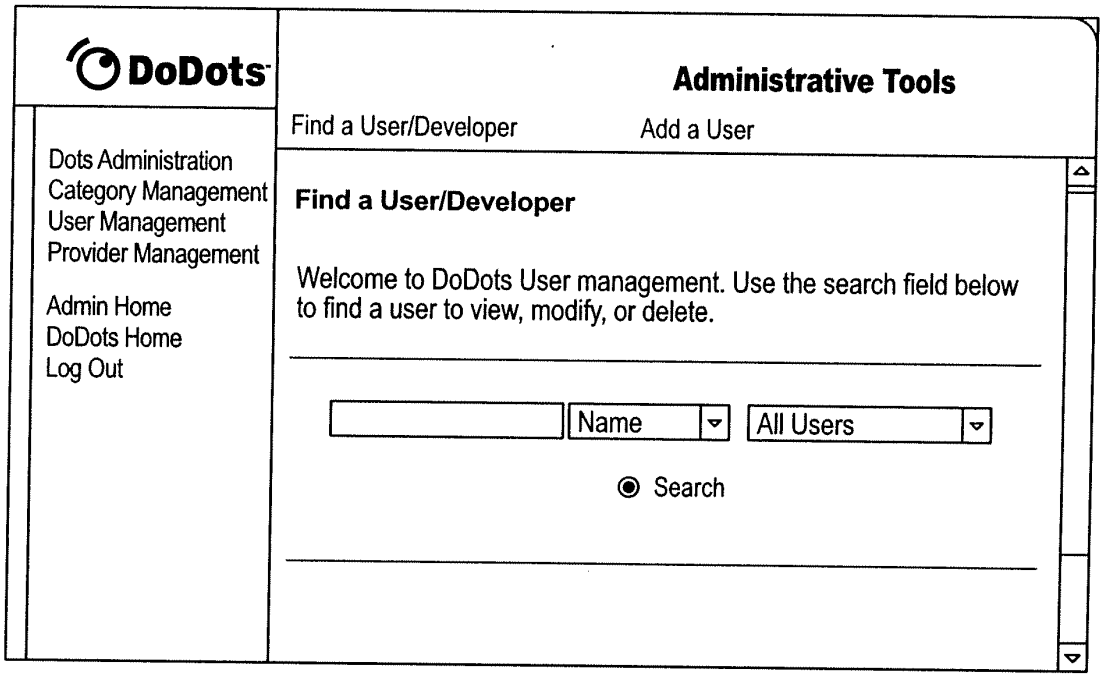

DoDots
Administrative Tools
Find a User/Developer
Add a User
Dots Administration
Category Management
User Management
Provider Management
Admin Home
DoDots Home
Log Out
Find a User/Developer
Welcome to DoDots User management. Use the search field below to find a user to view, modify, or delete.
Name
All Users
Search

FIG. 31A

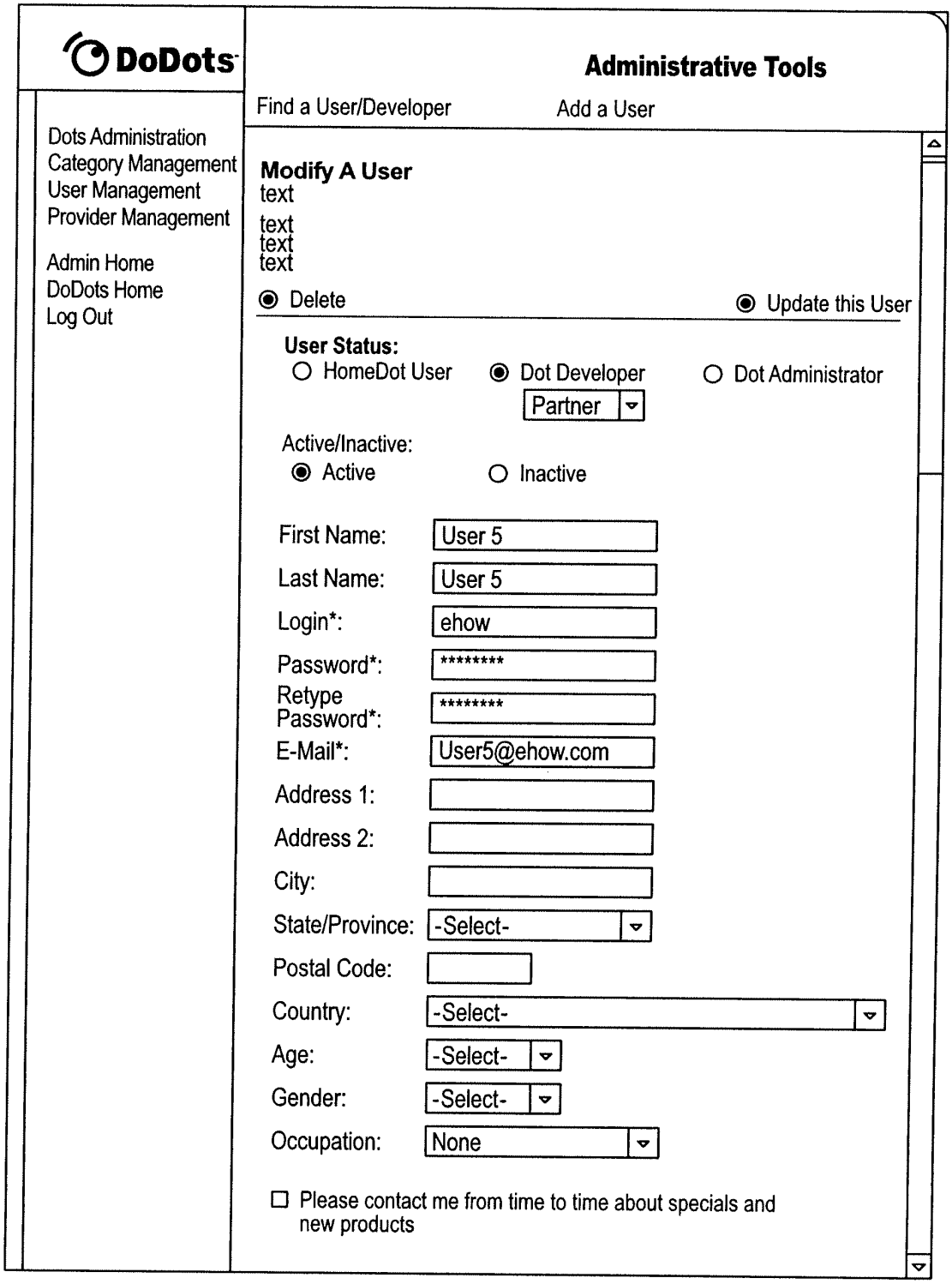

DoDots
Administrative Tools
Find a User/Developer
Add a User
Dots Administration
Category Management
User Management
Provider Management
Admin Home
DoDots Home
Log Out
Modify A User
text
text
text
text
Delete
Update this User
User Status:
HomeDot User
Dot Developer
Dot Administrator
Partner
Active/Inactive:
Active
Inactive
First Name:
User 5
Last Name:
User 5
Login*:
ehow
Password*:
****
Retype Password*:
****
E-Mail*:
User5@ehow.com
Address 1:
Address 2:
City:
State/Province:
-Select-
Postal Code:
Country:
-Select-
Age:
-Select-
Gender:
-Select-
Occupation:
None
Please contact me from time to time about specials and new products

FIG. 31B

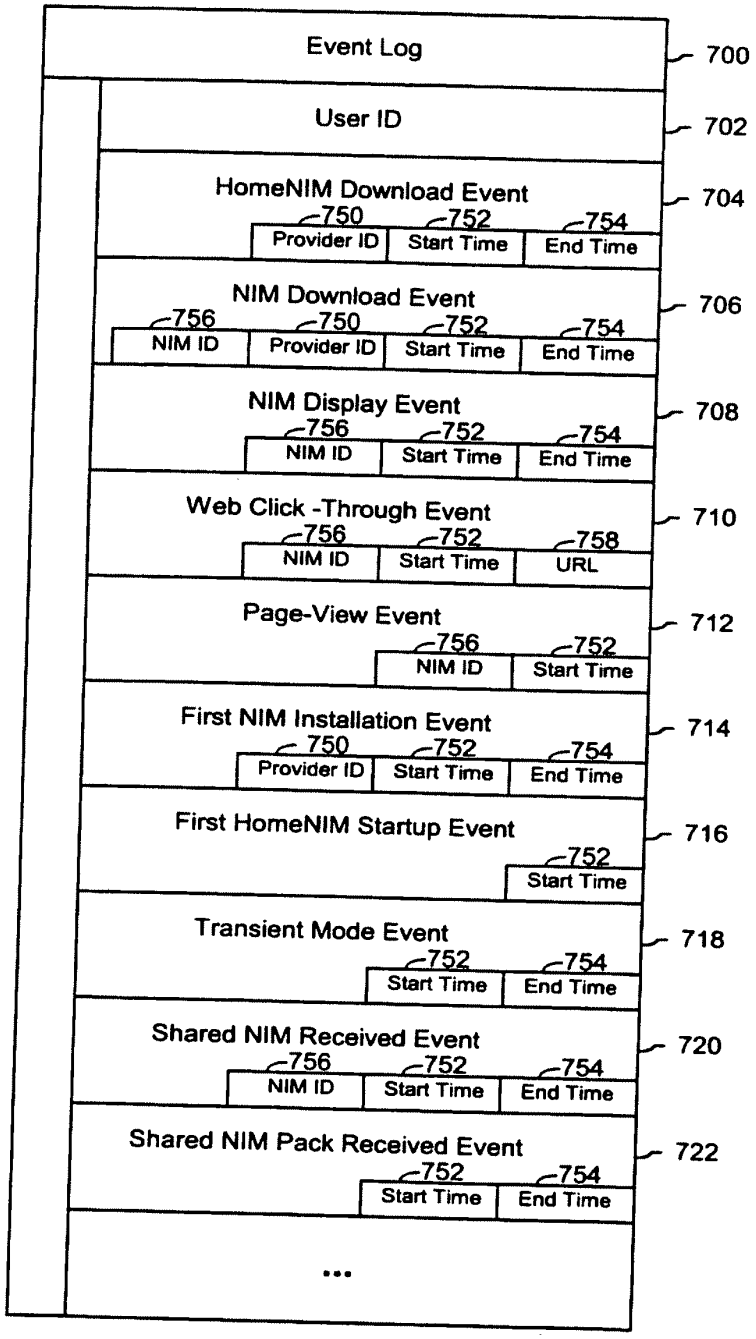

Event Log
700
User ID
702
HomeNIM Download Event
704
750
752
754
Provider ID
Start Time
End Time
NIM Download Event
706
756
750
752
754
NIM ID
Provider ID
Start Time
End Time
NIM Display Event
708
756
752
754
NIM ID
Start Time
End Time
Web Click -Through Event
710
756
752
758
NIM ID
Start Time
URL
Page-View Event
712
756
752
NIM ID
Start Time
First NIM Installation Event
714
750
752
754
Provider ID
Start Time
End Time
First HomeNIM Startup Event
716
752
Start Time
Transient Mode Event
718
752
754
Start Time
End Time
Shared NIM Received Event
720
756
752
754
NIM ID
Start Time
End Time
Shared NIM Pack Received Event
722
752
754
Start Time
End Time
...

FIG. 32

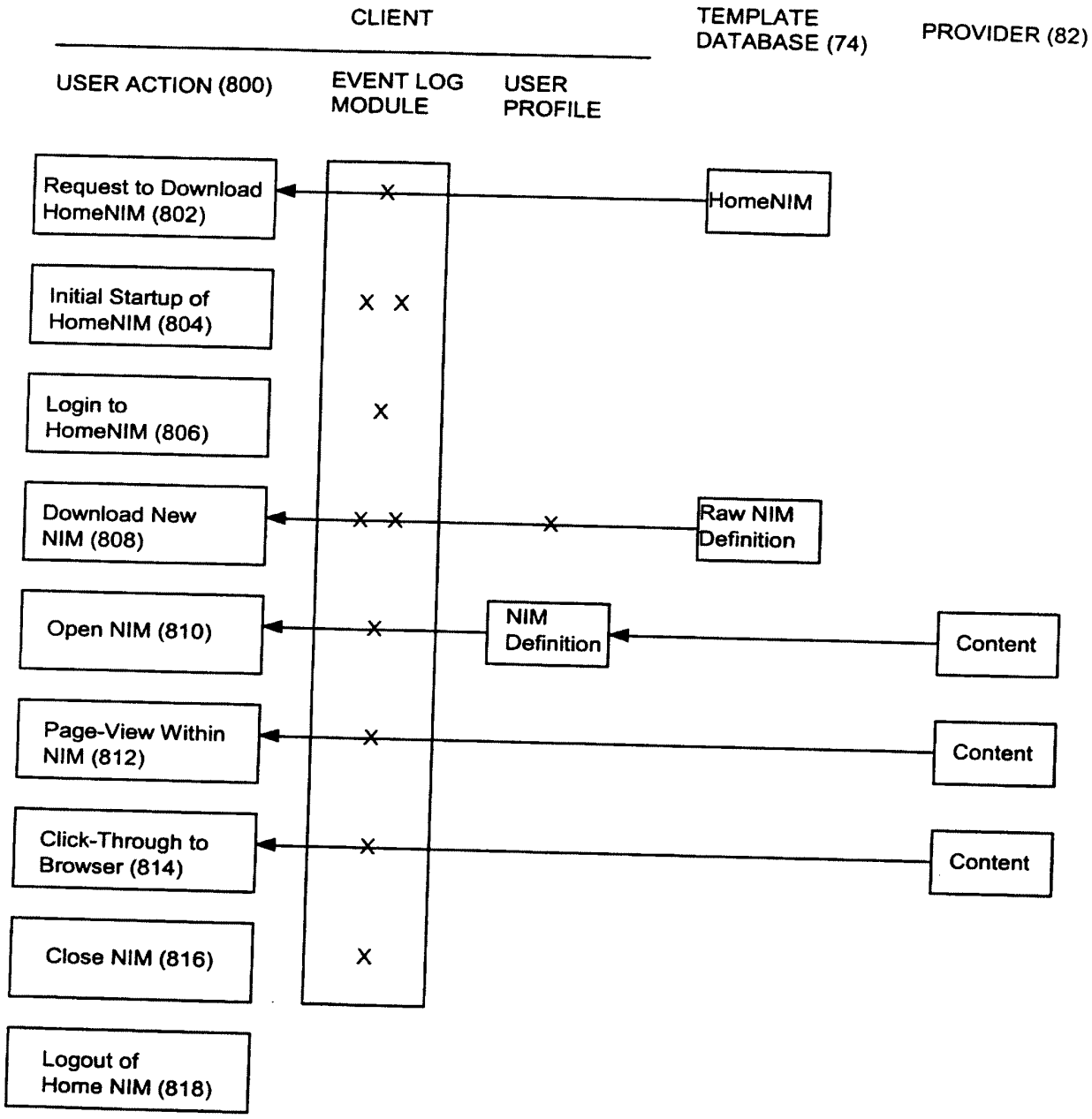

CLIENT
TEMPLATE DATABASE (74)
PROVIDER (82)
USER ACTION (800)
EVENT LOG MODULE
USER PROFILE
Request to Download HomeNIM (802)
HomeNIM
Initial Startup of HomeNIM (804)
Login to HomeNIM (806)
Download New NIM (808)
Raw NIM Definition
Open NIM (810)
NIM Definition
Content
Page-View Within NIM (812)
Content
Click-Through to Browser (814)
Content
Close NIM (816)
Logout of Home NIM (818)

FIG. 33

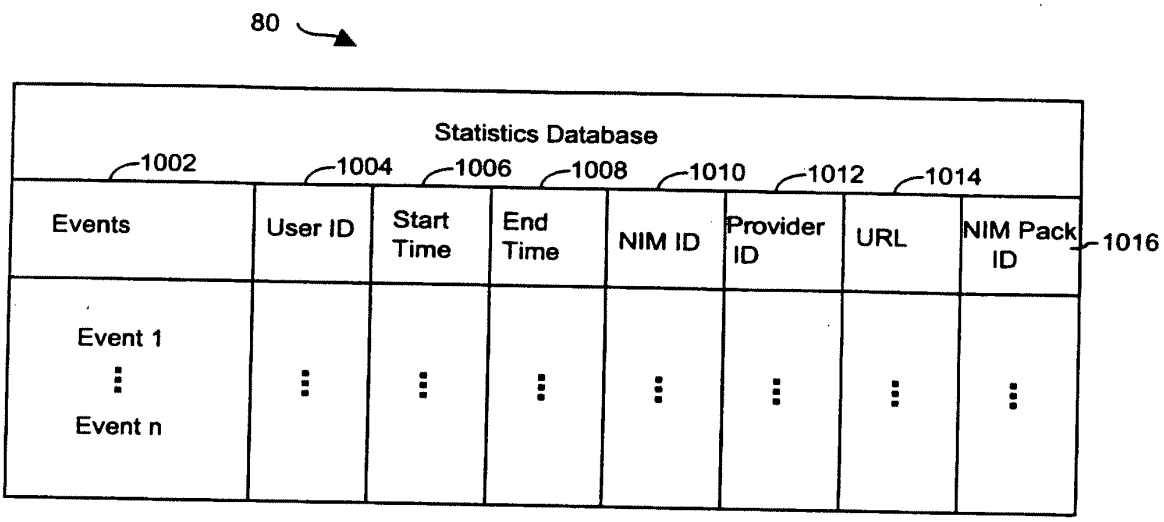

80
Statistics Database
1002
1004
1006
1008
1010
1012
1014
Events
User ID
Start Time
End Time
NIM ID
Provider ID
URL
NIM Pack ID
1016
Event 1
Event n

FIG. 34

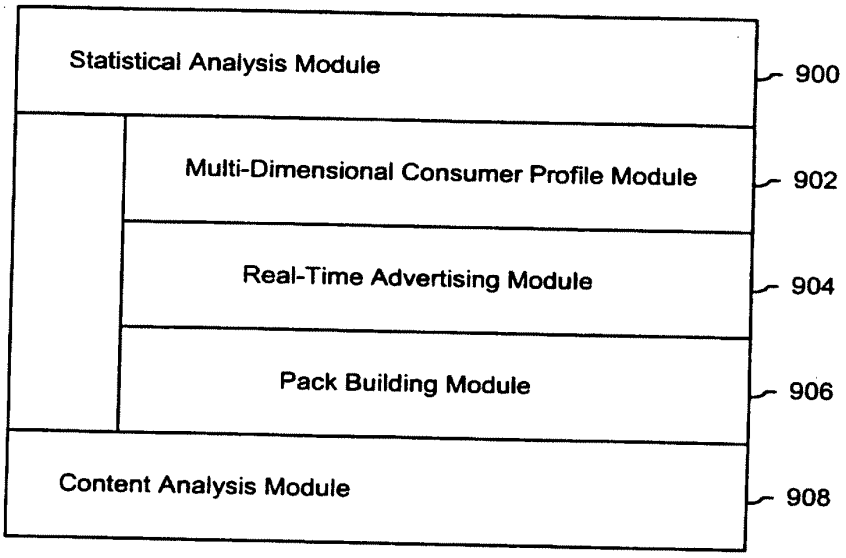

Statistical Analysis Module
900
Multi-Dimensional Consumer Profile Module
902
Real-Time Advertising Module
904
Pack Building Module
906
Content Analysis Module
908

FIG. 35

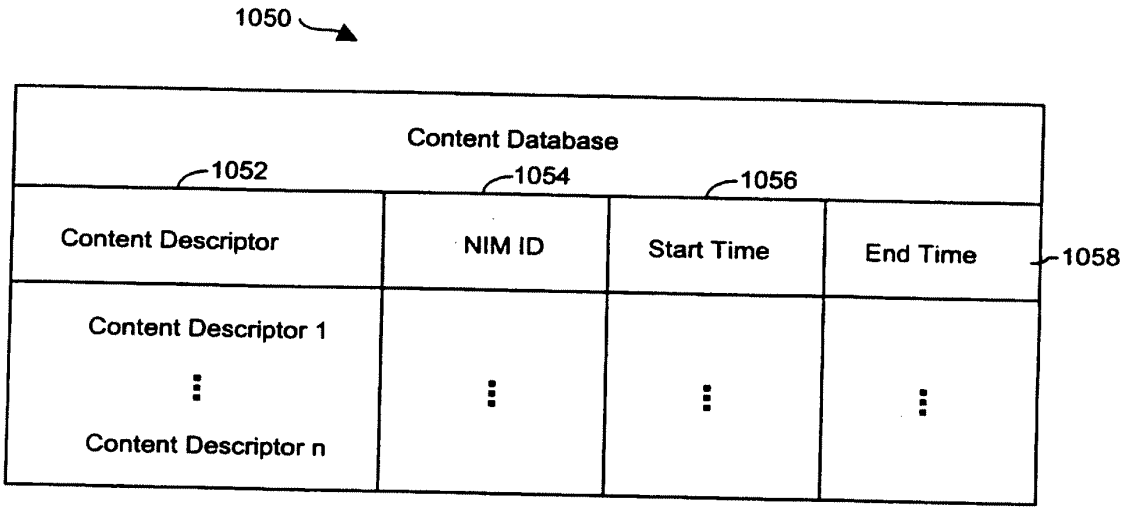

1050
Content Database
1052
1054
1056
Content Descriptor
NIM ID
Start Time
End Time
1058
Content Descriptor 1
Content Descriptor n

FIG. 36

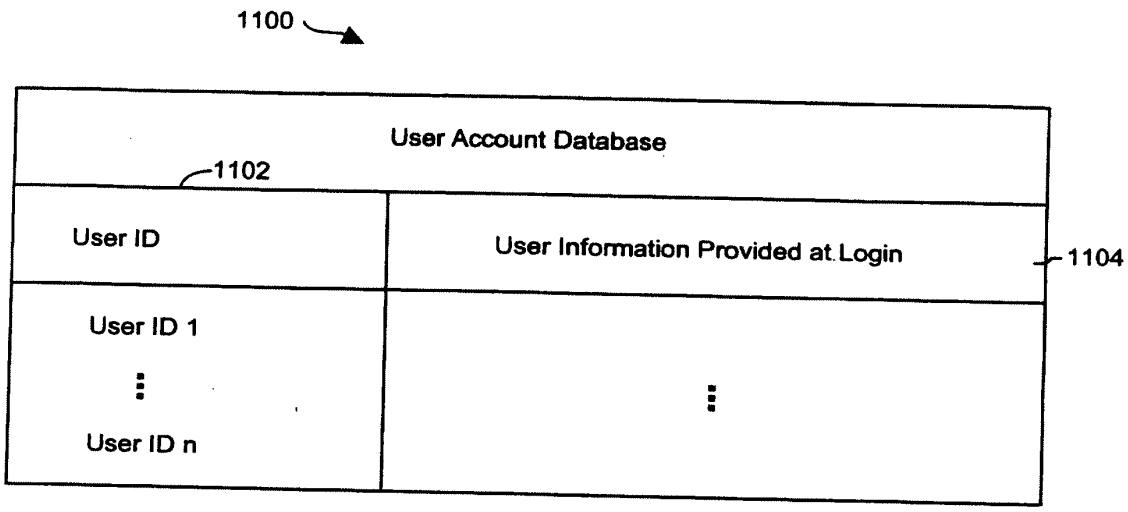

1100
User Account Database
1102
User ID
User Information Provided at Login
1104
User ID 1
User ID n

FIG. 37

SUBSTITUTE SPECIFICATION (MARKED UP)

DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] The present application is a continuation of and incorporates by reference U.S. Non-Provisional patent application Ser. No. 09/558,925, filed Apr. 26, 2000, now U.S. Patent No. 7,660,868, which claims priority from and incorporates by reference U.S. Provisional Application ser. nos. 60/131,083, filed Apr. 26, 1999, 60/131,114, filed Apr. 26, 1999, 60/131,115, filed Apr. 26, 1999, 60/176,687, filed Jan. 18, 2000, and 60/176,699, filed Jan. 18, 2000. The present application claims priority to U.S. Non-Provisional patent application ser. no. 09/558,925, filed Apr. 26, 2000, now U.S. Patent No. 7,660,868, and each of the aforementioned applications to which it claims priority.

[0002] The present application is also related to and incorporates by reference the following U.S. patent applications: Non-Provisional Application ser. no. 09/558,922, filed April 26, 2000, now U.S. Patent No. 7,756,967; Non-Provisional Application ser. no. 09/558,923, filed April 26, 2000; Non-Provisional Application ser. no. 09/558,924, filed April 26, 2000, now U.S. Patent No. 7,356,569; Non-Provisional Application Ser. No. xx/xxx,xxx, filed October 31, 2007 titled "Component For Accessing And Displaying Internet Content" (attorney docket no. IMS 07-05-01); Non-Provisional Application ser. no. 11/932,286, filed Oct. 31, 2007, titled "Server Including Components For Accessing And Displaying Internet Content And For Providing Same To A Client"; Non-Provisional Application ser. no.

11/932,392, filed Oct. 31, 2007, titled "Method For Accessing And Displaying Internet Content"; Non-Provisional application Ser. No. 11/932,427, filed Oct. 31, 2007, titled "Component For Coordinating The Accessing And Rendering Of An Application Media Package"; Non-Provisional Application ser. no. 11/932,456, filed Oct. 31, 2007, titled "Tracking and Tracing User Activity with Application Media Packages"; Non-Provisional Application ser. no. 11/932,585, filed Oct. 31, 2007, titled "System and Methods for Creating and Authoring Internet Content using Application Media Packages"; Non-Provisional Application ser. no. 11/932,630, filed Oct. 31, 2007, titled "Methods of Obtaining Application Media Packages"; Non-Provisional Application ser. no. 11/932,663, filed Oct. 31, 2007, titled "Indexing, Sorting, and Categorizing Dots"; Non-Provisional Application ser. no. 11/932,692, filed Oct. 31, 2007, titled "System and Methods of Messaging between Application Media Packages; and, Non-Provisional Application ser. no. 11/932,763, filed Oct. 31, 2007, titled "Component For Accessing And Displaying Internet Content In Association With a Web Browser Application".

BACKGROUND OF THE INVENTION

(03) A user operating a client computer typically accesses the Internet by using a viewer application, such as a browser to view web content provided at a destination address, typically a web page. In this context, web content and web applications are designed to fill the entire web page. It is known to divide the web content into different regions of a single web page. For example, personalized web pages can be specified, such that a user views a variety of content sources in a single page, such as stock information, weather information, and sports information, which is aggregated at the server that delivers the web page to the

user, who then views the aggregated content in a single web page. Observe that even when disparate content is aggregated, in this manner, it is reassembled into a full web page and is served through a full-screen browser. Web content and application developers therefore have limited control over the user experience: content is typically trapped within the frame of the browser. A developer's only alternative to engaging a user page-by-page in a browser is to develop, distribute, and support custom client software. In the Web browser scenario, it is the content provider, not the user that aggregates the information that is viewed by the user. Thus, the user is not in a position to separately aggregate the content at a client computer, instead the user is constrained to view the content that has been delivered in the manner provided by the server computer hosting the web page. There is a growing desire for individual users to fully control the aggregation and presentation of content and web applications that appears on a client computer.

(04) A user who wishes to view multiple web pages or applications can open multiple instances of a browser. However, the user will not be able to view each "full-screen" page at the same time. Instead, the user must adjust the windows corresponding to each browser instance and view only part of each page. The information appearing in each browser is not designed for viewing in this manner. Thus, the user cannot create an optimized display of content from multiple sources.

(05) Currently, content providers and end users have limited tools to alter the browser in which content appears. That is, the controls associated with a browser are not fully configurable. Thus, the vendor of a browser is in a position to brand the browser and regulate the controls associated with the browser. There is a growing desire for content providers to not only fill a browser with their content, but to also fully brand and control the

frame in which the content appears. Further, in some instances, content providers desire to limit the controls associated with a browser or viewer, so that a user is more inclined to view a single set of content, for example, by having limited access to previously viewed content.

(06) At the present time, it is relatively difficult to trace the content viewing activity of a client computer. In other words, it is difficult to identify the type of content that a particular user of a client computer favors. Consequently, there are limited tools 25 available to provide a user with tailored information that would be of particular interest to the user.

(07) In view of the foregoing, there is a need in the art to provide a technique for accessing multiple instances of distributable computer readable media in their entirety simultaneously, where these instances are typically smaller than the full pages used in 30 current web pages and web applications. There is a further need for providing the user with flexibility in selecting, collecting, relating and viewing such computer readable media, and for giving the media provider flexibility in directing media to a specific user and controlling the framework in which media is presented. Finally, there is a need to gather more accurate information regarding the type of content that a user enjoys, so that the user can be automatically provided with this content.

SUMMARY OF THE INVENTION

(08) The invention includes a method of presenting distributable computer readable media to a user in response to a user request. The method comprises the steps of identifying a definition of a Networked Information Monitor (NIM). A NIM frame is defined for the

NIM using the definition. Content is then retrieved for the NIM. Then, the content is placed in a NIM viewer defined by the frame.

(09) The invention also includes a method of altering a Networked Information Monitor (NIM). The method includes the step of receiving a message at a NIM. The message specifies a configurable feature of the NUM. The NIM is altered in accordance with the configurable feature of the message.

(10) The apparatus of the invention includes a computer readable memory to direct a computer to function in a specific manner. The computer readable memory includes a first executable module to identify a definition of a Networked Information Monitor (NIM). A second executable module defines a NIM frame for the NIM using the definition. A third executable module retrieves content for the NIM. A fourth executable module places the content in a NUM viewer defined by the frame.

(11) The apparatus of the invention further includes a computer readable memory with a first executable module to receive a Networked Information Monitor (NIM) message. The NIM message specifies a configurable feature of a NIM. A second executable module alters the NIM in accordance with the configurable feature of the NIM message.

BRIEF DESCRIPTION OF THE DRAWINGS

(12) FIG. 1 illustrates a system for processing distributable computer readable media in accordance with one embodiment of the present invention;

(13) FIG. 2 shows exemplary processing steps used to initiate an instance of a home networked information monitor (NIM) using the apparatus of FIG. 1;

(14) FIG. 3A illustrates a screen logo in accordance with one embodiment of the present invention;

(15) FIG. 3B illustrates a login construct in accordance with one embodiment of the present invention;

(16) FIG. 4 illustrates a set of visual manifestations corresponding to a collection of NIMs, including a home NIM;

(17) FIG. 5 illustrates a visual manifestation of a home NIM as well as a visual manifestation of a reference NIM that references additional NIMs;

(18) FIG. 6 illustrates how a NIM, referenced by the reference NIM of FIG. 5, is added to a processed user profile in response to a designated keyboard entry sequence or mouse click;

(19) FIG. 7 illustrates how a NIM is shared with other users in accordance with an embodiment of the present invention;

(20) FIGS. 8A and 8B illustrate screen panels that facilitate the collection of the description of a set of designated NIMs into a pack;

(21) FIG. 9A illustrates how the relative position of visual manifestations that correspond to NIMs remains fixed when the visual manifestations are within a predetermined distance of each other;

(22) FIG. 9B illustrates a representative pack in accordance with the present 20 invention;

(23) FIGS. 10A, 10B and 10C illustrate how a set of visual manifestations corresponding to a collection of NIMs is aligned against a boundary when a user selects the visual manifestations and pushes them against the boundary;

(24) FIG. 11 is a flow chart of the steps taken by a client to give a user access to a NIM where the user "collects" the NIM, in accordance with one embodiment of the invention;

(25) FIG. 12 is a flow chart of the steps taken by a client to present a NIM to a user, where the NIM has been "collected" previously by the user, in accordance with one embodiment of the invention;

(26) FIG. 13 illustrates a data structure for a NIM definition, stored in the NIM application server's template database or user profile database;

(27) FIG. 14 illustrates NMA message routing between NIMs and the message interface in the client parser application;

(28) FIG. 15 is a diagrammatic illustration of an embodiment of a NIM Management Module utilized in accordance with an embodiment of the invention;

(29) FIG. 16 is a diagrammatic illustration of an embodiment of the NIM Templates database utilized in accordance with an embodiment of the invention;

(30) FIG. 17 is an illustration of a main NIMIndex Web page used in accordance with an embodiment of the invention;

(31) FIG. 18 is an illustration of a single NIMIndex category used in accordance with an embodiment of the invention;

(32) FIG. 19 is an illustration of a full description of NIM content provided in accordance with an embodiment of the invention;

(33) FIG. 20 is an illustration of a Web page displayed to the user once the user has clicked to collect the NIM;

(34) FIG. 21 is an illustration of the main home NIM graphical user interface used in accordance with an embodiment of the invention;

(35) FIG. 22 is an illustration of a "Get New NIM" graphical user interface that may be used in accordance with an embodiment of the invention;

(36) FIG. 23 is an illustration of a "More NIMs" graphical user interface representative of an embodiment of the invention;

(37) FIG. 24 is a diagrammatic illustration of the ShareLink database used in accordance with an embodiment of the invention;

(38) FIG. 25 is an illustration of a Share NIM's graphical user interface according to an embodiment of the invention;

(39) FIG. 26 is an illustration of the main DevZone Web page utilized in accordance with an embodiment of the invention;

(40) FIG. 27 is a partial view of a NIM modification web page utilized in accordance with an embodiment of the invention;

(41) FIGS. 28A to 28D are graphical user interfaces of development NIMs (DevNIMs) utilized in accordance with an embodiment of the invention;

(42) FIGS. 29A and 29B are illustrations of Administrative Zone (AdminZone) Web pages utilized in accordance with an embodiment of the invention;

(43) FIGS. 30A and 30B are also illustrations of Administrative Zone (AdminZone) Web pages utilized in accordance with an embodiment of the invention;

(44) FIGS. 31A and 31B are further illustrations of Administrative Zone (AdminZone) Web pages utilized in accordance with an embodiment of the invention;

(45) FIG. 32 illustrates an embodiment of an event log that may be used in accordance with an embodiment of the invention;

(46) FIG. 33 illustrates the tracking of events in an event log module in accordance with an embodiment of the invention;

(47) FIG. 34 illustrates a statistics database that may be used in accordance with an embodiment of the invention;

(48) FIG. 35 illustrates a statistical analysis module and a content analysis module that may be used in accordance with an embodiment of the invention;

(49) FIG. 36 illustrates a content database that may be used in accordance with an embodiment of the invention; and

(50) FIG. 37 illustrates a user account database that may be used in accordance with an embodiment of the invention.

DETAILED DESCRIPTION OF THE INVENTION

(51) The present invention discloses a technology that is capable of processing distributable computer readable media. Distributable computer readable media includes, but is not limited to, standard web content, such as HTML, dHTML, images, imbedded ActiveX and Java applications, JavaScript, CSS, Pen scripts, Streaming Media, and/or Flash. The present invention is advantageous relative to prior art systems and methods because it provides improved mechanisms for simultaneously interacting with several independent sources of distributable computer readable media, collecting references to such media, and sharing such references with other users. The ac disclosed technology is further advantageous because it provides improved systems and methods for on screen management of distributable computer readable media.

(52) In the system and method of the present invention, a user logs into a server by providing a login identifier to a login construct. The login identifier is used by the server to obtain an unprocessed user profile that corresponds to the user. The unprocessed user profile is

processed by the server to generate a processed user profile. Advantageously, this processing step allows for up-to-date refinement of the user profile. Up-to-date refinements include, for example, the addition of advertisements directed to the user based on one or more characteristics in the user profile. The processed user profile is delivered to the client computer associated with the user. The processed user profile includes references to the networked information monitors (NIMs). As used herein, the term networked information monitor or NIM refers to a fully configurable frame with one or more controls; the frame through which content is optionally presented. The fully configurable frame utilized in accordance with the invention stands in contrast to present web browsers, which are branded by the browser vendor and which have limited means by which to alter the controls associated with the browser.

(53) Attention is initially directed toward the home NIM of the invention, which coordinates the activities of all other NIMs that are accessed by a user. The home NIM facilitates much of the technology of the present invention, including the ability to simultaneously review multiple sources of distributable computer readable media as well as to package and distribute such media.

(54) FIG. 1 is a general illustration of a system in accordance with one embodiment of the present invention. In FIG. 1, a network 10 is operated in accordance with the present invention. Network 10 includes at least one user or client computer 20, at least one server computer of class 50, and optionally one or more server computers of class 82. User computer 20 as well as server computers of class 20 50 and 82 are each connected by transmission channel 44, which is any wired or wireless transmission channel.

(55) User computer 20 is any device that includes a Central Processing Unit (CPU) 24 connected to a random access memory 30, a network connection 28, and one or more user input/output ("i/o") devices 40 including output means 42. Output means 42 is any device capable of communicating with a user and includes, for example, a video monitor, a liquid crystal display, voice user interfaces, and/or integrated graphic means such as mini-displays present in web-phones. Typically, user computer 20 includes a main non-volatile storage unit 22, preferably a hard disk drive, for storing software and data. Further, user computer 20 includes one or more internal buses 26 for interconnecting the aforementioned elements. In a typical embodiment, memory 30 includes an operating system 32 for managing files and programs associated with user computer 20. In some embodiments, operating system 32 includes a registry 34 that has one or more references to specified locations in system 10. The exemplary memory 30 of FIG. 1 further includes a web browser 36 for viewing web content and a client parser application 38 for facilitating low level functionality, such as login and logout procedures, in accordance with the present invention. In some embodiments, client parser application 38 uses the one or more references in registry 34 to obtain a login construct from server 50. In various embodiments, in accordance with the present invention, client parser application 38 runs in conjunction with one or more software modules, such as an event log module 98, which tracks user activity, a message interface module 106, which serves as a communication interface between the client parser application 38 and web server 58 and/or external web servers, a home NIM 108, which references one or more NIMs 110, and a visual management system 114 which regulates the characteristics of visual manifestations of NIMs 108 and 110 when displayed on output device 42. Furthermore, in some

embodiments, client parser application 38 runs in conjunction with instances of web browser 36 as well as web server 58 as detailed below.

(56) Server computer 50 includes standard server components, including a network connection device 46, a CPU 52, a main non-volatile storage unit 54, and a random access memory 56. Further, server computer 50 includes one or more internal buses 48 for interconnecting the aforementioned elements. Memory 56 stores a set of computer programs, modules and data to implement the processing associated with the present invention.

(57) The embodiment of memory 56 illustrated in FIG. 1 includes a web server 58 for processing requests received from client computer 20. Web server 58 has many components, including a variety of modules and data structures to assist users that want to log into system 10. Namely, login module 60 handles an entry request from a client computer 20 and accepts a login identifier that corresponds to a user from client computer 20. Login constructor 62 generates a login construct in response to a call for a login construct and transfers the login construct to client 20. Login constructor 62 dynamically generates login constructs using updated login NIM content 64. Login validation module 66 works in conjunction with user profile database manager 100 to identify an unprocessed user profile, which is associated with a user provided login, in user profile database 76. If an unprocessed user profile corresponding to the user provided login does not exist in user profile database 76, login validation module 66 associates a new unprocessed user profile with the login identifier.

(58) When an unprocessed user profile is identified by login validation module 66, it is processed by user profile processor module 68 to produce a processed profile. It will be

appreciated that the services of user profile processor module 68 are highly advantageous because they allow for last minute user profile revisions. Such revisions include, for example, the addition or customization of NIMs referenced by the user profile, and/or server redirect information that is a function of current server load in system 10. Importantly, the processed user profile includes a reference to a home NIM. The home NIM is capable of accessing each of the NIMs that are represented in the processed user profile. Memory 56 further includes delivery module 70, which coordinates the delivery of portions of the home NIM to the client based on parameters specified in the processed user profile.

(59) Once a user has successfully logged into system 10, request server module 72 handles requests for specified NIMs from client 20. When such a request is received, request server module 72 routes the request to an address that corresponds to the specified NIM and transmits the specified NIM to client 20. One class of specified networked information handled by request server module 72 is requests for NIMs. When such a request is received, request server module 72 searches NIM templates database 74 for the specified NIM. NIM templates database 74 includes a large number of NIM templates. Each NIM template defines the characteristics of a specific NIM, including fully configurable frame characteristics, viewer and control characteristics, and NIM content references.

(60) The web server 58 illustrated in FIG. 1 further includes additional modules 102 to handle specialized features of the present invention. For example, one embodiment of the present invention provides a mechanism that allows users to distribute NIMs to each other. In such embodiments, a special server module 102 provides instructions for storing the NIMs, which are to be distributed, in sharelink database 78. Advantageously, NIMs that are distributed to other users are customizable. A user can, for example, resize and position a

particular NIM prior to sharing it with another user. Indeed, it is possible, in such embodiments, for a user to arrange a series of NIMs in a unique arrangement and then distribute the collection of NIMs in the designated NIMs in the designed arrangement. As an illustration, a user arranges a first NIM that represents a scrolling stock ticker at the bottom of an output means, such as a computer screen, a second NIM that tracks the NASDAQ top ten most heavily traded stocks in the upper left corner of the output means, and a third NIM that tracks headline news on the upper right hand corner of the output means. Then, the user distributes the three NIMs in this customized arrangement to other users. Observe that in this example a user of a client computer is aggregating separate sets of information in different NIMs. This stands in contrast to prior art approaches where a web server running on a server computer aggregates information in a single page.

(61) System 10 is highly scalable and thus supports a large number of users. This scalability stems from the fact that the server 50 is delivering the definition associated with a NIM. The content displayed in the NIM may be located on a separate computer.

(62) Memory 56 provides a statistical analysis module 104 for tracking key events associated with users. This information is stored in statistics database 80. The information collected by statistical analysis module 104 is used for a wide variety of purposes, including server load optimization and directed advertising, as discussed below. As described below, the statistical information gathered in accordance with the invention includes fully traced events defining the type of content and the duration over which all content is viewed by a user. This type of comprehensive information is not available using present techniques.

(63) Much of the distributable computer readable media that is available for processing is stored as content elements 94 on server 82. Server 82 is a standard web server that includes components such as a network connection device 88, a CPU 86, a main non-volatile storage unit 84, a random access memory (RAM) 92, and one or more internal buses 90 for interconnecting the aforementioned elements. RAM 92 includes some of the content elements 94 stored by server 82. Other content elements 94 are stored in storage unit 84. In some embodiments, a single web server 58 is capable of directly accessing content elements 94 located on one or more servers 82. In other embodiments, each server 82 has a resident web server module that works in conjunction with server 50 to identify, optionally dynamically generate, and serve content elements 94 upon demand.

(64) Now that general architecture of a system in ac dance with the present invention has been disclosed, attention turns to FIG. 2, which discloses a method for logging into system 10 (FIG. 1). In the first processing step shown in FIG. 2 (202), a user initiates a session on system 10 by requesting the global login script "session_config." It will be appreciated that the term "session_config" merely provides an illustrative name for the global login script and that the technology of the present invention is by no means limited to this name or the script described.

(65) The request for "session_config" originates on client 20 and is sent to server 50 where it is processed by login module 60 of web server 58. Upon receiving request 202, login module 60 creates a "session_config" global login script (204). Processing step 204 is advantageous relative to systems that have static global login scripts because it allows for the incorporation of highly variable information. This highly variable information includes, for example, system settings such as up-to-date server redirect information, server content

address changes, directed advertisements, and messages. An exemplary "session_config" is found in Example 1 below. Each line of data has an associated numeral. The remaining text in the example describes the purpose of select data.

(66) EXAMPLE 1

(67) Version tag that identifies the latest home NIM version

(1) </SESSION_CONFIG VERSION="alpha:3"

(68) Upgrade event that is sourced when home NIM version is outdated

(2) <UPGRADE=http://www.NIM.com /QuickOpen.exe>

(69) LOCATION OF SERVER-SUPPORTED FUNCTIONALITY Default base address for server supported functionality

(3) <METHODS BASEURL=http://neo.NIM.com /servlet/NIMServer/>

(4) <ADD_USER URL="addUser/>

(5) <GET_USER URL="getUser"/>

(6) <SET_USER URL="setUser/>

(7) <GET_SES_CONFIG URL=http://www.NIM.com /home_NIM/s_cnfg.xml"/>

(70) Server-based functionality for setting password

(8) <SET_PASSWORD URL="setPassword"/>

(71) List of all publically available NIMs

(9) <GET_MASTER_NIM_LIST URL="getMasterNIMList"/>

(10) <GET_ALL_CONFIG URL="getAllConfig"/>

(11) <SET_ALL_CONFIG URL="setAllConfig/>

(12) <SET_ALL_STATS URL="setAllStats"/>

(13) <GET_NIM_TEMPLATE URL="get_NIM_Template"/>

(72) Location of server-side NIM and pack sharing functionality

(14) <ADD_SHARE URL="addShare"/>

(15) <GET_SHARE URL="getShare"/>

(16) <AUTH_TEST URL="DOeCHO?AUTH=TRUE"/>

(73) Redirect information

(17) <DO_REDIR URL="doRedir"/>

(18) </METHODS>

(74) Flexible content layer that defines default NIM frame appearance, including the default appearance of the frame of a home NIM

(19) <FRAMES>

(75) Default NIM frame appearance

(20) <NIM>

(21) <IMAGES BASEURL=

http://www.NIM.com /home_NIM/NIM_FrameImages/>

(22) </NIM>

(76) Default home NIM frame appearance

(23) <HOME_NIM>

(24) <IMAGES BASEURL=

http://www.NIM.com /home_NIMImages/>

(25) </HOME_NIM>

(26) </FRAMES>

(77) Location of system NIM templates

(27) <NIMs>

(28) <ADD TEMPLATE="http://www.NIM.com / . . . /add_NIM_XML.xml"/>

(29) <HELP TEMPLATE="http://www.NIM.com / . . . /help_NIM_XML.xml"/

(30) <LOGIN TEMPLATE="http://www.NIM.com / . . . /login2.xml"/

(31) </NIMs>

(32) </SESSION_CONFIG>

(78) Line 1 of the exemplary "session_config" of Example 1 provides the version tag for the expected version of the home dot system that corresponds to the "session_config" script. In one embodiment, client parser application 38 determines whether it is up-to-date using the information in line 1. If client parser application 38 determines that it is outdated, an upgrade request is made in accordance with the instructions provided by the UPGRADE flag of line 2. In one embodiment, the UPGRADE flag in line 2 of Example 1 describes the location of an executable program, one of skill in the art will appreciate that this flag can in fact reference any form of instruction, including a flat file, a web page, a script, a symbol, or an address.

(79) Lines 3 through 18 in Example 1 define the functionality that is provided by a server, such as server 50. For example, line 8 of Example 1 provides the location of a set of instructions that are called when a user requests a password change. Furthermore, line 9 of Example 1 provides the location of master list of NIMs that are publically available. Lines 14 and 15 of Example 1 provide the location of specialized server-side functionality that allows users to share data such as NIM definitions.

(80) Lines 19 through 26 of Example 1 define where the default appearance of a NIM and a home NIM are found within system 10. Lines 27 through 31 define a collection of system NIMs. A system NIM is any type of NIM that is to be distributed to each user of system 10.

In some embodiments, system NIMs are used to provide a core functionality. In Example 1, line 28 defines a NIM that provides users with a convenient mechanism for collecting additional NIMs. Line 29 defines the location of a NIM that is invoked when the user presses a help button associated with a home NIM. Finally, line 30 defines the location of a NIM that is used to log into system 10.

(81) Returning to FIG. 2, once login module 60 has created "session_config," it is sent back to requesting client 20 (206). When a "session_config" is received by client 20, client parser application 38 parses the global login script in order to identify a reference to a login constructor 62 (208). Login constructor 62 is a server-based module that generates a construct that allows a user to log into system 10. When client parser application 38 locates the reference to login constructor 62 in "session_config," a request for a login construct is directed to the identified reference (210). In Example 1 above, the reference to the login construct is provided in line 30. On line 30, the global variable "LOGIN TEMPLATE" is assigned the URL address "http://www.NIM.com . . . /login2.xml." Client parser application uses the URL assigned to the global variable "LOGIN TEMPLATE" to make a request for a login constructor 62 that is directed to this URL. When login constructor 62 receives a request for a login construct, it generates a login construct (212).

(82) Login construct 148 (FIG. 3B) illustrates a type of login construct that is generated in one embodiment of the present invention during processing step 212. Before the login construct is executed on client 20, a schematic such as logo 146 (FIG. 3A) is displayed on output means 42. As illustrated in FIG. 1, login constructor 62 is a component of web server 58. However, there is no requirement that login constructor 62 be a component of web server 58. In fact, login constructor 62 is a standalone software program in some

embodiments of the present invention whereas in other embodiments, login constructor 62 is merely a script, such as a PERL script, that is processed by an interpreter program native to server 50. In still other embodiments, login constructor 62 is merely a simple flat file that includes a set of instructions that are interpretable by client parser application 38. In such embodiments, login constructor 62 is the login construct. In embodiments in which a login construct is dynamically generated, it is possible to introduce last minute changes in the login construct. Thus, an advantage of the exemplary login process shown in FIG. 2 is that there are multiple stages in which updated information is used to customize the login process based on the environmental variables.

(83) Once a login construct has been prepared by login constructor 62, it is transferred back to client 20 (214) (FIG. 2) and executed in conjunction with client parser application 38 (216). The login constructs of the present invention are a form of NIM. Therefore, one function of processing step 216 is to obtain the login NIM content 64 (FIG. 1) specified by the login construct from server 64. In login construct 148, for example, the login NIM content includes the shape and functionality of "Exit button 160," message 150, the shape and functionality of "New user" button 152, the functionality of "Forgot it?" button 154, and login panel 156. When processing step 216 is completed, the user uses the login construct to provide a login identifier (218).

(84) In FIG. 3B, a user has provided the login identifier "Galliani." The definition of login identifier as used in the present invention is to be broadly construed. In some embodiments, login identifiers include a unique name and a corresponding password. In other embodiments, a login identifier does not have a password. This is particularly the case when the user is a guest or a new user and there is no user profile associated with the user.

(85) Working in conjunction with client parser application 38, the login construct accepts the user login and sends it to server 50 for validation (218). As illustrated in the exemplary system of FIG. 1, web server 58 includes a login validation module 66 to verify the login identifier provided by user (220). Typically, processing step 220 involves a look-up operation in which the login identifier is used to query user profile database 76 for an unprocessed or raw user profile that matches the login identifier. In embodiments that include a password, validation step 220 includes a password verification step. Successful completion of processing step requires entry of a valid login identifier sequence in processing step 218. When processing step 220 has been successfully completed, the raw or unprocessed user profile corresponding to the login identifier is obtained from user profile database 76 (FIG. 1) (222) and is processed by user profile process module 68 to produce a processed or finalized user profile that is delivered to client 26 (226). In sonic embodiments, a user profile 76 includes user contact information, such as the name, address, telephone number and email address of a user. Additionally, some embodiments of system 10 provide different types of access privileges. For example one embodiment of the present invention includes developer access privileges, administration access privilege, and general user access privileges. In such embodiments, the access privileges that have been granted to a user are stored in the user profile 76 associated with the user.

(86) The processed user profile includes a reference to each NIM in system 10 that is associated with the login identifier provided in processing step 216. One of the NIMs referenced by the processed user profile is the home NIM that corresponds to the login identifier provided in processing step 216. When executed in conjunction with client parser application 38 in processing step 226, the home NIM provides a mechanism for accessing

each of the NIMs referenced by the processed user profile. Like the login construct, the home NIM includes several components, including pull down menus and screen manipulation functionality. The reference to the home NIM in the processed user profile includes the system 10 address of each of these components. Therefore, in one embodiment, construction of the home NIM in processing step 226 involves one or more requests to server 50 and/or server 82 for content (228) that is then rendered (230) in accordance with the home NIM description provided in the processed user profile. In some embodiments, the home NIM is distinct from other NIMs in the sense that a large proportion of the home NIM in such embodiments is pre-compiled. Such embodiments are advantageous because some of the functionality provided by the home NIM requires substantial client 30 processing resources. Therefore, to minimize such processing resource requirements, many aspects of the home NIM are pre-compiled in some embodiments. In other embodiments, however, the home NIM has a structure that is substantially the same as a regular NIM. In such embodiments, simple script commands are used to identify the NIM as a home NIM.

(87) Upon completion of processing step 230, the user is granted access to all of the technologies of the present invention, including the ability to view multiple NIMs simultaneously, collect new NIMs, customize NIMs, and share customized NIMs with other users. An exemplary processed user profile is provided in Example 2. Once again, each line of data is identified with a numeral, while the remaining text in the example describes select data. In some embodiments, the user is granted specific privileges and the extent to which the user is granted access to system 10 is regulated by the types of privileges that have been granted to the user.

(88) EXAMPLE 2

(1) SAMPLE PROCESSED USER PROFILE

(2) <ALL CONFIG>

(89) NIMs AND PACKS THAT CORRESPOND TO THE USER

(3) USER

(90) NIM definition 1

(4) <NIM DOMAIN="ZDNet" GLOBALID="1" KND="news"

(5) <FRAME BACKGROUNDCOLOR=#FFFF00" COLLAPSED="FALSE"

(6) FIXHEIGHT="TRUE" FIXWIDTH="TRUE" NAME="ZDNet Breaking News"

(7) PIXELHEIGHT="275" PIXELWIDTH="235" X="RIGHT" Y="TOP">

(8) <TITLE COLOR=#000000" JUSTIFY="RIGHT" TEXT=" "/>

(9) <TITLEBARIMAGE DOWN=http://www.NIM.com / . . . /feed/titlebar.gif

(10) HOVER=URL address to a first GIF file <param 1> . . . <param N>

(11) INACTIVE=URL address to a second GIF file <param 1> . . . <param 2>

(12) NORMAL=URL address to a third GIF file <param 1> . . . <param 2>

(13) <BOTTOMBARIMAGE DOWN=

http://www.NIM.com / . . . /feed/bottombar.gif

(15) HOVER=URL address to a fourth GIF file <param 1> . . . <param 2>

(16) INACTIVE=URL address to a fifth GIF file <param 1> . . . <param 2>

(17) NORMAL=URL address to a sixth GIF file <param 1> . . . <param 2>

(18) </FRAME>

(19) <MENU/>

(20) <CONTROL_LAYOUT HEIGHT="1" HEIGHTSCALES="TRUE" WIDTH="1"

(21) WIDTHSCALES="TRUE"> <CONTROL CLASS="Browser" HEIGHT="1"

(22) ID="1" KIND="A" LEFT="0" TOP="0"

(23) URL=http://www.mandala.com /cgl/zdnet/zdfeedl.cgi WIDTH="1"/>

(24) </CONTROL_LAYOUT>

(25) <CATEGORIES/>

(26) <EVENTS/>

(27) </NIM>

(91) NIM definition 2

(28) <NIM DOMAIN=NIM DOMAIN 2 GLOBALID=`2`

(29) </NIM>

(92) NIM definition N

(30) <NIM DOMAIN=NIM DOMAIN 2 GLOBALID=`N`

(31) </NIM>

(93) Pack definition 1

(32) <PRESET TITLE="New DotPack">

(33) <NIM GLOBALID="1" X="RIGHT" Y="TOP"/>

(34) <NIM GLOBALID="2" X=RIGHT Y="320"/>

(35) </PRESET>

(36) </SHARE>

(94) Last state of the home NIM

(37) <LASTSTATE>

(38) <PRESET TITLE=" ">

(39) <NIM GLOBALID="1" X="RIGHT" Y="TOP"/>

(40) <NIM GLOBALID="2" X=RIGHT Y="280"/>

(41) </PRESET>

(42) <HOMENIM COLLAPSED="FALSE" HEIGHT="134" X=616" Y="109"/>

(43) </LASTSTATE>

(44) </ALL_CONFIG>

(95) Example 2 describes a representative processed user profile in accordance with the present invention. In general, a processed user profile includes three major components: (i) a definition of each NIM associated with the user, (ii) a description of each pack associated with the user and, (iii) the last state of each home NIM associated with a user. In Example 2,

the definition of each NIM associated with the user is found on lines 4 through 31. Specifically, lines 4 through 31 describe NIM definitions I through N. In Example 2 there is only one pack associated with the user. This pack, entitled "New DotPack," is found on lines 32 through 35 of Example 2. The final major component of the processed user profile found in Example 2 is the last state of the home NIM, which is defined on lines 37 through 43. This code stores the last state of the home NIM. Such last state information includes whether the home NIM was collapsed, and the position of the home NIM on the screen.

(96) When the user wishes to log out of system 10, the processed user profile is transferred from client 20 to server 50. When web server 58 receives the processed user profile, it passes the processed user profile to user profile database manager 100. User profile database manager 100 stores the processed user profile as the unprocessed user profile 76 corresponding to the user. In some embodiments, such a storage operation involves a conversion process. For example, advertisements or specific system NIM definitions are stripped from the processed user profile in order to convert the processed user profile to the unprocessed user profile 76 that corresponds to the user. In some embodiments, the processed user profile is periodically transferred, in its entirety or incrementally, from client 20 to server 50 and saved in the manner described in the log out procedure above. Such timed periodic or event based backup procedures are possible because NIM definitions are efficiently described, thus the absolute size of a processed user profile remains relatively small. Accordingly, timed backups of a processed user profile to user profile database 76 are possible without extensive use of system 10 bandwidth or server 50 resources.

(97) At this stage, a system (FIG. 1) and a login procedure (FIG. 2) in accordance with the present invention has been disclosed. Although the system and login procedure was discussed using an example where only one home NIM was associated with a user, it will be appreciated that, in some embodiments, any number of distinctly different home NIMs are associated with a user. Furthermore, a user can simultaneously execute multiple instances of a particular home NIM on client 20 or, indeed, any number of different home NIMs. In one embodiment, a developer or merchant provides a user with a highly customized home NIM that provides specialized functionality. In such embodiments, the user collects the home NIMs and, therefore, a processed user profile includes a description of more than one home NIM.

(98) Attention now turns to some of the advantages and features of the present invention. In FIG. 4, a visual manifestation of the home NIM 162 is illustrated. One advantage of the home NIM, which is an advantage that is common to NIMs in general, is that the content of the NIM is not trapped in a third party viewer. In fact, the home NIM definition regulates the actual appearance of the home NIM. The home NIM definition is formed by general parameters and commands found in "session_config" as well as customized parameters and commands in the processed user profile. The division of the home NIM definition between a system level file and a user level file represents a balance in the tension between the need for a system 10 host to insure a consistent level of quality, through the proper implementation of general parameters and commands, and the desire of each user to create highly customized home NIMs. Lines 20 through 22 of Example 1 provide an example of general parameters that are defined in "session_config." Lines 23 through 25 define the source location of home NIM frame images. In home NIM 162 (FIG. 4), such home NIM

frame images include the image used to represent buttons 164 through 174, and menu tabs 1 d 178. Furthermore, lines 20 through 22 of Example 1 define the location of other images that are used to construct default NIMs. An example of user initiated home NIM customization is found in lines 33 through 39 of Example 2, which define a "LASTSTATE" definition for the home NIM, including the dimensions of the visual manifestation corresponding to the home NIM on line 38 (HEIGHT="134" X="616" Y="109") and indicates that the home NM is not collapsed upon startup (COLLAPSED="FALSE").

(99) The visual manifestation of home NIM 162 illustrates additional benefits and features of a home NIM in accordance with the present invention. When a user selects tab 176, a list of the NIMs that are present in the processed user profile associated with the user is displayed in viewer 180. As disclosed in more detail below, a user has the option to associate a collection of NIMs into an object termed a "pack". The pack references some subset of the NIMs associated with a user as well as associated state information. This arrangement includes, for example, whether a visual manifestation corresponding to each MM is displayed on output means 42 or not, the dimensions of each visual manifestation, and the position of each visual manifestation. The name of each pack is stored in the processed user profile. A user reviews packs associated with the user by selecting tab 178 (FIG. 4). In FIG. 4, the user only has one pack, "Customized DotPack" 182. When the user selects pack 182, each NIM in the pack is restored in accordance with the state information stored in the pack definition.

(100) In total, FIG. 4 represents a typical visual experience provided by one embodiment of the present invention. In addition to home NIM 162, visual manifestations 184 and 186, corresponding to two additional NIMs in the processed user profile, are displayed. Visual

manifestation 184 provides functionality that allows a user to manage an address book, schedule appointments, or create groups and plan activities. Visual manifestation 186 represents a NIM that provides time and date information.

(101) FIG. 5 shows the visual manifestation of home NIM 162 of FIG. 4 with tab 176 selected. Accordingly, each of the NIMs in the processed user profile associated with the user is listed in list 188. The user can activate any of the listed NIMs by clicking on the NIM name. In addition to the NIMs in list 188, home NIM 162 includes core NIMs that are defined in "session_config." In the "session_config" of Example 1, cores are found on lines 28 and 29. Specifically, line 28 provides the address of an XML-based definition for the add template functionality associated with button 172 in FIGS. 4 and 5, and line 29 provides the address of an XML-based definition for the help template functionality associated with button 174 in FIGS. 4 and 5.

(102) Importantly, the user can categorize NIMs using filter 190. Categories include such topics as sports, personal, weather, etc. Furthermore, the user can add NIMs to the processed user profile associated with the user as well as delete NIMs. There are a variety of mechanisms that enable a user to add a NIM to the processed user profile. One mechanism is to receive links to NIMs from other users of system 10 (FIG. 1), as disclosed below. Another mechanism is to toggle button 172 in order to activate a visual manifestation associated with NIM 192 (FIG. 5).

(103) NIM 192 provides a system that enables users to add select NIMs to their user profile with a single click or keystroke sequence. NIM 192 includes tab 194 that allows the user to select premiere NIMs and a general tab 196 that allows the user to review a general catalog of NIMs that is present in NIM templates database 74 (FIG. 1). In one embodiment, when a

user selects a NIM in list 198 (FIG. 5), the NIM is added to list 188 and is incorporated into the processed user profile associated with the user. In this way, the user can collect NIMs of interest to the user using a single mouse click. By illustration, consider the case in which a user selects the NIM "AnyDay Calender" in list 198. In response to this selection, a definition of the NIM "AnyDay Calender" is obtained from NIM templates database 74 and is copied directly into the processed user profile associated with the user. Furthermore, the title of the selected NIM, "AnyDay Calender" is added to list 188. Finally, a visual manifestation that corresponds to the NIM "AnyDay Calender" is displayed on output means 42 (FIG. 1). As a result, the display illustrated in FIG. 5 adopts the appearance illustrated in FIG. 6.

(104) In FIG. 6, the NIM "AnyDay Calendar" appears at the top of list 188. Furthermore, a control 101 associated with the NIM "AnyDay Calendar" in list 188 is filled, indicating that the NIM is currently active. Additionally, as illustrated in FIG. 6, a visual manifestation 103 corresponding to NIM "AnyDay Calendar" appears on the output means. The user has the ability to toggle this NIM between an inactive and active state by selecting control 101.

(105) In one embodiment, the user is provided with the option of (i) incorporating a NIM selected in list 198 into the processed user profile or (ii) transiently executing the NIM on client 20. Furthermore, when the user receives NIMs from other users, the user has the option to transiently operate the received NIMs on client 20. If the user decides to keep the transient NIMs at a later date, the user has the option to add the transient NIMs to the processed user profile at that time. Thus, in such embodiments, the user effectively has the option to "preview" NIMs before adding them to the processed user profile. This is advantageous because it reduces the chances of filling the user profile with undesirable

NIMs. Such a feature is particularly advantageous in the case of novice or inexperienced users of system 10. Furthermore, one of skill in the art will appreciate that the concept of transient NIM execution raises the possibility of executing NIMs on a client 20 during a period of time in which the user is not logged into system 10. For example, consider a NIM that is executed on a client 20 after a user initiated response to a web page advertisement presented in web browser 36. Although the user is not logged into web server 58 and therefore does not have a processed user profile resident on client 20, the user can execute the NIM on client 20 on a transient basis. Furthermore, if the user wishes to add the transiently executed NIM to the user profile 76 associated with the user, the user can log into web server 58 and then add the NIM to the processed user profile that is delivered to client 20 as a function of the log in process.

(106) Another important feature of the present invention is the ability for users to share NIMs with each other. For example, if a user wishes to share the NIM "AnyDay Calendar" that was added to list 188 in FIG. 6, the user clicks "share" button 170 (FIG. 6). In response, panel 105 is displayed (FIG. 7). Because "share" button 170 is pressed while tab 176 is active in the illustration provided by FIGS. 6 and 7, panel 105 lists each of the NIMs associated with the user. If, however, "share" button 170 is pressed while tab 178 is active rather than tab 176, panel 105 will list each of the packs associated with the user instead of each of the NIMs. Returning to the situation illustrated in FIG. 7, the user shares a NIM with other users by selecting the NIM to be shared from list 107 and then toggling button 109 "Share via email." In one embodiment, the user has the option to select multiple NIMs from list 107 using predefined keystroke operations. For example, in one embodiment, the user selects multiple NIMs by clicking on several of the NIMs in list 107 with a mouse

button while depressing the "shift" button on the keyboard. When a user decides not to share a NIM and panel 105 is displayed, the user presses cancel button 111 and panel 105 is dismissed.

(107) When a user toggles "share via email" button 109 at a time when one or more NIMs in list 107 have been selected, the definition of each selected NIM is copied from the processed user profile associated with the user into a container and the container is sent to server 50 (FIG. 1). In the embodiment shown in FIG. 1, the container is received by web server 58. Web server 58 includes instructions for routing the container to sharelink database 78 where the container is stored. When the container is stored, a unique identifier is assigned to the container. Although a large number of different mechanisms for generating a unique identifier are practiced in accordance with this aspect of the invention, in one embodiment, the unique identifier assigned to the container upon storage in sharelink database 78 can be subsequently processed to form a URL address that specifically references the container within the context of system 10. In one embodiment, after a unique identifier has been assigned to the container, an e-mail program is launched on client 20 and the user is requested to designate the recipients of the designated NIMs. Then, each recipient is provided with the unique identifier associated with the container in an e-mail message. When the recipient clicks on the unique identifier, a call is made for a copy of the associated container from sharelink database 78 and the container is delivered to the client 20 associated with the recipient.

(108) As is readily apparent upon review of FIG. 7, the user has the option to size and position the visual manifestation that corresponds to each NIM. Furthermore, by toggling controls, such as toggle button 101 (FIG. 7), the visual manifestation of a NIM is toggled between an

on state and an off state. Such functionality is highly advantageous. First, by using this functionality, the user has the option to create unique arrangements. Second, NIM developers have the ability to control the default position and size of NIMs as well, and can therefore produce an arrangement of NIMs to further specialized purposes. Finally, because the NIMs of the present invention are not trapped in third party applications that have a set of undesirable features such banner ads, the utility and overall appearance of an arrangement of NIMs is enhanced and adopts an independent value. Using the technology disclosed in the present invention, the user collects an assortment of NIMs and arranges them in a customized fashion. The user has the option to "capture" favored arrangements into constructs known as packs, which have been briefly discussed previously.

(109) FIGS. 8 and 9A illustrate the formation of a pack using the arrangement of NIMs illustrated in FIG. 4. The process begins when the user toggles button 164 "Make Pack" in FIG. 4. In the embodiment illustrated by FIGS. 8 and 9A, panel 113 (FIG. 8) is displayed when the user toggles button 164 (FIG. 4). Panel 113 advises the user to open and arrange each of the NIMs that are to be included in a pack. In the case of FIG. 4, for example, such an arrangement could include the arrangement of NIMs 184 and 186. The user indicates that specified NIMs are in a desired arrangement by selecting button 115 "Next" (FIG. 8). When button 115 is toggled, prompt 113 is terminated and prompt 117 is displayed to prompt the user for a name to associate with the designated pack. The user indicates that a name 119 has been provided for the pack by selecting "Done" button 121.

(110) In the embodiment shown in FIG. 8, the user further has the option to return to panel 113 and rearrange the specified NIMs before committing to pack creation by selecting the "Back" button 123. In the situation illustrated in FIG. 8, the user has provided the name

"New DotPack." FIG. 9A illustrates the state of the visual manifestation corresponding to home NIM 162 after the user has selected "Done" button 121 (FIG. 8). Specifically, the name "New DotPack" is added to pack list 125 and tab 178 is activated to display the user pack list rather than the user NIM list that is displayed when tab 176 is activated. Furthermore, in response to the user selection of "Done" button 121 in FIG. 8, a reference to each NIM specified by the user is collected into a pack, along with some state information, and the pack is stored in the processed user profile associated with the user. Representative state information for each NIM stored in a pack includes whether the NIM was collapsed and the position of the NIM. In some embodiments, the state information includes the dimensions of the last visual manifestation corresponding to the NIM to have been displayed on output means 42.

(111) FIG. 9 illustrates pack 139, which is delineated with a dashed box. Pack 139 includes five NIMs 133. Each NIM 133 includes two primary components, a viewer 135 for viewing content and a frame 137 for providing user functionality. Each viewer 135 provides a platform for reviewing machine readable information, such as HTML, dHTML, images, imbedded ActiveX and Java applications, JavaScript, CSS, Streaming Media, and/or Flash. Furthermore, in some embodiments, viewer 135 provides functionality for processing digitally recorded sound. Observe then that each NIM of the invention utilizes machine readable information that is easily retrieved from a specified address. If the content provider of this information desires to change the machine readable information, changes can be made and then delivered the next time that the machine readable information is addressed. This stands in contrast to prior art techniques in which updates to executable code can be relatively difficult to distribute.

(112) Each frame 137 provides functions and controls for manipulating the visual manifestation of the NIM 133 corresponding to the frame. For example, some frames 137 include a dismiss button 141. When the user clicks on dismiss button 141, the corresponding NIM 133 is dismissed. Additionally, some frames 137 include a menu button 143. When the user clicks on button 143, a menu appears. In one embodiment, the menu is a pull down menu and the menu items are defined in the corresponding NIM definition. In an abstract example, the NIM definition provides a menu definition of the format:

Menu 143-(I)-"Get more NIMs" URL

(113) where (I) refers to the N.sup.th element of the menu that is activated when menu button 143 is pressed, "Get more NIMs" is the menu element name, and URL is the event or web address that is called when the user selects the N.sup.th element of the menu associated with button 143.

(114) The developer has full control over all aspects of the appearance and functionality of NIM frame 137. Thus, a NIM developer has the ability to control, for example, the functionality located at any corner of frame 137, functionality placed along the top or bottom of the frame, or on the sides of the frame. As an illustration, frame 137-3 includes a control button 145 that allows the user to resize the visual manifestation of NIM 133-3. One of skill in the art will appreciate that the bottom row of NIM frame 137 could have any number of buttons, including a forward control, a backward control, and similar controls, each of which provides the user with distinct functionality.

(115) An advantage of the present invention is that it is possible to embed commands that affect frames 137 in the content of the computer readable media delivered to frame viewer

135. The use of embedded commands provides NIM developers with powerful tools and additional flexibility. For example, a developer can use embedded commands, such as the menu command above, to design a NIM that has a context sensitive help menu. Each time a particular event occurs, the developer changes the content of the help menu using embedded commands. For example, when a sound file is delivered to a NIM, the sound file may be attached to a set of embedded commands that provide sound manipulation functionality in the form of a pull down menu. Elements of such a pull down menu include a command for saving the sound file to disk, commands for fast forward, stop, and play, and commands for sound enhancement. After the sound file has been played, embedded commands are used in this example by the developer to reset the menu associated with button 143 to some default state. One of skill in the art will appreciate the benefits and advantages of a frame 137 that is capable of being modified based on commands embedded in the machine readable media delivered to the corresponding viewer 135. The developer can use presentation tools, such as adding transient help buttons, resizing the visual manifestation of the NIM, changing the frame border color, changing the title of the frame, and changing the frame border patterns, to create a more effective application.

(116) One of the advantages of the disclosed pack system is that it provides a convenient mechanism for rapidly assembling NIMs that track diverse sources of information. Furthermore, when a useful set of NIMs is collected into a pack, the user can share the pack with other users using the same procedure previously identified for sharing one or more NIMs. It will be appreciated that in some embodiments, NIMs are commercial applications and that appropriate use of packs provides an additional dimension for application development. Therefore, in some embodiments, pack recipients are subscribers to a service

provided by a pack developer. In other embodiments, pack recipients are purchasers or licensees of packs. In still other embodiments, pack recipients receive packs developed by friends, family members, or business associates.

(117) The technology of the present invention further provides a set of NIM management tools to help a user manage displayed NIMs. In some embodiments of the present invention, these management tools are provided by visual management module 114 (FIG. 1). Two such NIM management tools are, in fact, properties that are associated with NIMs, namely magnetism and snapping. The property of magnetism is exhibited when a visual manifestation corresponding to a first NIM is dragged or moved near a visual manifestation corresponding to a second NIM. When this occurs, the two NIMs exhibit a magnetism that causes the first NIM to accelerate toward the second NIM. However, when the visual manifestations are within a predetermined distance of each other, the NIMs snap together. In one embodiment, the predetermined distance that triggers the two visual manifestations to snap together is a gap of about five pixels. While the above discussion describes the principles of snapping and magnetism based on a pair of NIMs, there is in fact no limitation on the number of NIMs that can be snapped together and furthermore, the principle of magnetism is not dependent on whether a NIM is in fact snapped to another NIM or not.

(118) An additional management tool, illustrated in FIG. 9A, provides a mechanism for selecting multiple NIMs and for moving the NIMs in a coordinated fashion. In FIG. 9A, visual manifestations 162, 184 and 186 corresponding to respective NIMs are locked together. In response, halo 127 is drawn around the selected locked visual manifestations to graphically notify the user which NIMs are locked together. As mentioned previously, the user has the option to position NIMs as a coordinated group. For example, in one

embodiment, when halo 127 is displayed and the user moves mouse arrow 129 after selecting one of the visual manifestations corresponding to a locked NIM, a target manifestation 131 is displayed to indicate to the user the position that the selected NIMs will be relocated to if the user clicks a mouse key. In some embodiments, target manifestation is a shadow image of the NIMs within halo 127 rather than the box depicted in FIG. 9.

(119) The present technology further provides additional methods for controlling visual manifestations of NIMs. For example, in one embodiment, the user has the option to select multiple NIMs by pressing a predefined key such as the keyboard "Ctrl" key, before selecting a specified NIM. While continuing to depress the control key, the user has the option to select additional NIMs and add the corresponding NIMs to a group. The user then has the option to move each of the NIMs in a single coordinated fashion as a group. Furthermore, by entering a designated keyboard or mouse sequence, the user has the option to move a single NIM even in situations where the NIM is in a group. In one embodiment in accordance with this aspect of the invention, the user clicks the visual manifestation corresponding to a locked NIM that the user wishes to move in an independent manner and the user does not click the visual manifestation when the user wishes to move the NIM in a manner that is coordinated with the other NIMs. Additional features of the present invention include the option to select rows or columns of NIMs using specialized control sequences. For example, in one embodiment of the present invention, the user selects a column of NIMs by clicking on a NIM while holding down the alphanumeric character "c" on the keyboard.

(120) It will be appreciated that one advantage of the present invention is that it is possible to display multiple NIMs and that each NIM provides a specialized visual experience. Therefore, NIM alignment tools are advantageous because they allow users to quickly make room on output means 42 for additional NIMs and/or to produce highly styled NIM arrangements. Accordingly, the present technology provides a specialized feature to rapidly align NIMs. This technology is illustrated in FIG. 10. The technology works in conjunction with the tools for selecting multiple NIMs. In FIG. 10A, the user selects the visual manifestations 147, 149 and 151 corresponding to respective NIMs using, for example, the column select feature disclosed above. Then, the user pushes the selected NIMs against boundary 153. In one embodiment, boundary 153 is the horizontal or vertical edge of output means 42. In another embodiment, all visual manifestations corresponding to NIMs are displayed in a single viewport such as a window. In such embodiments, the horizontal and vertical edges of the window each represent a boundary. FIG. 10B illustrates how visual manifestations 147, 149, and 151 are automatically aligned when they are pushed against a boundary, such as boundary 153. In some embodiments, the relative alignment between the visual manifestations is preserved even after the manifestations are moved in subsequent action by the user. The present technology further allows for the rearrangement of NIMs along a particular axis. For example, if NIMs are substantially oriented along a vertical axis as shown in FIG. 10B and the user wishes to realign the NIMs on the horizontal axis, all the user has to do is push the collection of NIMs against a horizontal border. For example, when the user pushes visual manifestations 147, 149 and 151 against border 155, the visual manifestations realign to conform to border 155 thus resulting in the view depicted in FIG. 10C.

(121) The features of the home NIM of the invention have been fully described. Attention presently turns to the architecture and operation of individual NIMs utilized in accordance with the invention.

(122) In one embodiment of the invention, after a user has logged into the system, as discussed above in connection with FIGS. 1 to 10, the user interface 40 displays the home NIM 162 as shown in FIG. 5. The home NIM typically includes a list of NIMs 188, referred to in FIG. 5 as "MyDots." These are NIMs which have been "collected" by the user. The list of collected NIMs, along with their associated definitions, is stored on the server in the user profile database 76, and downloaded from the application server 50 in response to a request from the client parser application 38. The local copy of the processed user profile is then further processed when the user collects or uses NIMs.

(123) Collected NIMs may be opened or closed by clicking on the control button next to the NIM's name or on the NIM's name itself, in list 188, and all NIMs may be closed by clicking on the "all off button` 166. The user may place NIMs into categories in a list of categories 190, which can be edited by clicking on the-Edit" button 168. New NIMs may be added to the user's collection of NIMs by clicking on the "Get" button 172.

(124) FIG. 5 also shows a NIM 192 with a list of NIMs 198, which may be previewed and/or collected by the client 20. The user may preview or collect a NIM by clicking on the associated name of the NIM, e.g., "eHOW". The steps taken to provide the NIM to a user are shown in FIG. 11. After logging in (step 240) the user clicks on the name of a NIM, and the client parser application 38 sends a request including the NIM_ID of the NIM definition, to the applications server 50 via the transmission channel 44 (step 241). Alternatively, the user may click on a NIM link before logging in, for example if the link

has been e-mailed to the user, and then, after clicking on the link, log in. In another aspect, the user could view, but not collect, a transient NIM without ever logging in.

(125) After the user is logged in and has clicked on the NIM, the applications server 50 retrieves the NIM definition from the NIM template database 74 using the NIM ID, in step 242, and provides it to the client 20 in step 243. The client 20 receives the NIM definition from the applications server 50 in step 244, and the client parser application 38 creates a frame in the display of the user interface 42 in step 245. In step 246, the client 20 requests the necessary content elements 94 stored at the URLs identified in the NIM definition from the corresponding content server 82. The content server 82 transmits the content 94 in step 247, and in step 248 the client parser application 38 places the content in the viewer, which is enclosed by the frame, allowing the user to preview the NIM. Alternatively, the client parser application 38 may simply collect the NIM, adding it to the user's processed user profile.

(126) The user may then view the NIM on the user interface display 42, and may interact with the NIM much in the same way as a user may interact with Internet content or web applications. This may change the NIM from its present, "raw" state to a used state reflecting alteration or use of the NIM by the user. For example, the user may direct the NIM to different content within the NIM if the NIM content enables the user to do so. Or, the user may provide information to the content server 82 which allows the NIM to be personalized. The user may additionally be given the option of changing the size of the frame.

(127) If the user collects the NIM, the NIM will be added to the user's list of collected NIMs such as the list 188 shown in FIG. 5. Additionally, the client parser application will add the

NIM's definition to the processed user profile, and, on logout, send the processed user profile to the application server 50. Thus, the NIM's "state" will be preserved. Alternatively, the client parser application may collect the NIM automatically, without waiting for a user command, by adding the NIM definition directly to the processed user profile.

(128) If the NIM's state has been altered by the user or by the content--if for example, the user has directed the NIM to Internet content other than the initially-displayed content, provided personalizing information, or changed the properties of the frame, or if the content itself has caused an alteration in the NIM--this alteration will be reflected in the NIM definition stored in the user profile database 76. Information which personalizes the resulting content, instead of being stored in a "cookie" on the client's hard drive, can be stored as part of the NIM definition. This advantageously permits personalization of content, such as web content that is associated with the NIM content and the user, without storing a cookie on the client 20.

(129) A user may also access a NIM which has been previously collected, and possibly altered by use as explained above. As previously described, the user profile 76 includes NIM definitions for NIMs which have been viewed and collected by each user. A screen shot showing NIMs 188, which have been previously collected by a user is shown in FIG. 5. The steps taken to provide the user with NIMs which have been previously collected are shown in FIG. 12.

(130) As discussed above, on login (step 250) the user's profile is retrieved by the client parser application 38 in the client 20 (step 251 and 252). The user's profile, stored in the user profile database 76, includes the NIM definition for each of the NIMs previously collected,

and possibly altered, by each user. The NIM definitions, as discussed above, includes the NIM frame definition and the definition of the controls for filling the viewer within the frame with content. After log in, a local copy of the processed user profile is stored on the client 20, and this copy is further processed as the user collects new NIMs, or uses new or collected NIMs such that the NIMs are altered.

(131) When the user clicks on the name of a collected NIM (step 253) the client parser application 38 creates a frame in the display 42 of the user interface 40 in step 254. At step 255, the client 20 requests the necessary content elements 94 stored at the URLs identified in the NIM definition from the corresponding content servers 82, which provide the content 94 in step 256. It will be appreciated that these URLs need not be the same as the initialization URLs in the "raw" NIM definition stored in the NIM template database 74 on applications server 50, and in fact the content servers need not be the same content servers corresponding to the initialization URLs. In step 257, the NIM parser application 38 places the content in the NIM frame, and the NIM is then fully opened.

(132) FIG. 13 illustrates a data structure for a NIM definition. As discussed above, a NIM is defined as a frame that contains a collection of controls, or functional units, such as a web rendering control or a GIF rendering control. The NIM frame surrounds a viewer, which displays the addressed content. The MM has a defined layout or arrangement of the controls, and defined initialization input data, e.g. data and URLS, for each control or element, in the NIM. NIM definitions are available to the client parser application via NIM links. The NIM links "point" to NIM definitions, which include all the information needed to build a NIM frame and fill the NIM with NIM content. Thus, NIMs links are easily collected, associated into packs, and shared by users.

(133) In one embodiment, the NIM definitions are defined using Extensible Markup Language (XML), so that the NIM as a whole--the frame and the content within the viewer--is advantageously as flexible as standard web content. NIMs are extremely flexible, because the definition of the NIM is content, rather than compiled code. The NIM definition defines the structure of the NIM, and everything that is visible in a NIM is based on standard Internet content, such as HTML, dHTML, or GIFs, and is referenced or pointed to by the NIM definition. An "application"-type NIM, such as a web calendar or web mail, may be changed by the user, by the content provider, or by other content, while advantageously avoiding the need to distribute and support a hard-coded compiled application. The definition of a NIM thus includes everything that is needed for the NIM to be rendered and filled with Internet content.

(134) As shown in the exemplary embodiment of FIG. 13, the definition of a NIM includes tags that identify the NIM 270, define and configure the NIM frame 271, specify and layout the controls 273 in the NIM viewer, and specify parameters to initialize all the NIM's components with content or data.

(135) In one embodiment, a NIM is identified by three ID strings 270: GlobalID, Domain and Kind. A GlobaliD is used when the MM definition is within a share. It is unique with respect to other NIM tags in the share. A NIM's domain is a unique label for the owning company or developer of the NIM, such as "dodots.com ." Finally, a NIM's kind, which is specified by the NIM's developer, is a helpful identifier for finding the NIM, but need not be unique. Examples of possible NIM kinds include "mp3", "scriblets," and "calculator." As discussed above, a NIM definition will typically be written in a format which facilitates sharing of data over the Internet, such as XML. An XML specification for the NIM

identification strings, for one embodiment of the invention follows. The bold text identifies NIM definition data, while the remaining text describes the data.

GLOBALID="string" Used only within <SHARE> tags. This GLOBALID must be unique with respect to other <NIM> tags in this <SHARE>.

DOMAIN="string"

(136) Unique label for the owning company of this NIM. In theory, NIMs may be limited to communicating with NIMs only from their own domain.

KIND="string"

(137) Helpful identifier for finding such a NIM from another NIM. Does not have to be unique.

(138) The NIM definition also includes the definition of a frame 271, which specifies the frame size and shape, and optionally the frame orientation and/or location on the user's screen. The space within the frame is the control space or viewer; visible controls are distributed within the control space or viewer.

(139) The NIM definition may optionally include controls for: a titlebar; a NIM menu with flexible menu entries; an exit button; and a bottombar. A typical layout for these components is: titlebar at the top of the control space, with menu on the left and exit button on the right, and the bottombar at the bottom.

(140) The titlebar component gives the user a place to grab and drag the NIM in a windowed environment. In one embodiment, it is implemented as a GIF rendering control that can be

targeted to a local or remote titlebar image. The titlebar will preferably have a fixed height and width that is a function of the NIM's width. The titlebar is preferably capable of being located at any position on the periphery of the NIM. Overlay text can also be specified to layer on top of the titlebar image. The bottombar may be implemented in a similar fashion, but typically will not include text overlay. The titlebar and bottombar may be filled in with initialization data from a fixed data file, or alternatively with Internet content from, example, an initialization URL.

(141) In one embodiment, a menu definition 271 is also included in the NIM definition. The menu includes items and actions of the NIM provider's choosing. For example, menu items may include the title "browse" associated with the action of targeting a full-screen browser or another NIM, and retrieving content for that browser or NIM from a specified address such as a URL. Logging off, or directing the NIM to another address or URL, are also possible menu action items. Menu action items that require communication of messages between the NIM and another NIM may also be provided--for example, opening another NIM, or changing the content of another NIM that is already open. Communication of messages between different parts of the system is discussed below.

(142) An XML specification for a frame, titlebar, bottombar and menu, for one embodiment of the invention follows:

<FRAME>

<TITLE>

TEXT="string"

JUSTIFY=-"LEFT"|"CENTER"|"RIGHT"

COLOR="#XXXXXX"

PIXELWIDTH="integer"

Width in pixel units. Overrides WIDTH attribute.

PIXELHEIGHT="integer"

Height in pixel units. Overrides HEIGHT attribute.

WIDTH="integer"

Width in NIM units. Default value is 1.

HEIGHT="integer"

Height in NIM units. Default value is 1.

X="integer"|"LEFT"|"CENTER"|"RIGHT"

Initial X position in screen coordinates. Default is center.

Y="integer"|"TOP"|"CENTER"|"BOTTOM"

Initial Y position in screen coordinates. Default is center.

FIXWIDTH="TRUE"|"FALSE"

Default is false.

FIXHEIGHT="TRUE"1"FALSE"

Default is false.

BACKGROUND COLOR="#XXXXXX"

Default is white.

<TITLEBARIMAGE>

JUSTIFY="LEFT"|"CENTER"|"RIGHT"

TILELEFT="integer"

TILERIGHT="integer"

NORMAL="URL"

DOWN="URL"

HOVER="URL"

INACTIVE="URL"

<BOTTOMBARIMAGE>

JUSTIFY="LEFT"|"CENTER"|"RIGHT"

TILELEFT="integer"

TILERIGHT="integer"

NORMAL="URL"

DOWN="URL"

HOVER="URL"

INACTIVE="URL"

<MENU>

Contains zero or more <ITEM> tags.

<ITEM>

TITLE="string"

TOOLTIP="string"

ICON="URL"

ID="string"

Must be unique.

<ACTION> RECIPIENT="address" MESSAGE="string"

(143) As shown in FIG. 13, the NIM definition also includes layout and definition of the controls 273. A control may be visible and render some sort of visual or text display, either static or dynamic. A control may be hidden, for example a functional element that is not necessarily visual such as a Java control. The control definition 273 includes identification of the types of controls, the layout of the controls, and initialization information. In one embodiment, NIM controls are specified and identified by class, kind and ID. Class defines the type of NIM control and is not unique. Kind is a useful identifier selected by the developer, and again is not unique. The NIM ID is unique within a user's processed profile.

(144) Different classes of controls may be used. For example, a control may be a web rendering object, which can render web content such as HTML, dHTML, images, imbedded ActiveX and Java applications, JavaScript, CSS, Streaming Media, and/or Flash. Alternatively, a control may be any object capable of rendering any kind of computer readable media, such as a GIF rendering object or an custom-designed object to display a particular kind of information. Alternatively, a control may be an object capable of processing any kind of application logic, such as a Java module. For example, an on-line brokerage firm could implement a custom stock-chart-rendering control, and define a NIM to use this control.

(145) As discussed above, the control space is located within the frame, and one or more controls can be flexibly positioned within the control space, and these controls may include a titlebar and a bottombar, as well as other controls. The location of controls is specified by the layout in the definition of the controls 273 within the NIM definition. In one embodiment, the controls are laid out or positioned within the NIM frame according to a flexible grid. In this embodiment, the NIM definition allows the control space to be

subdivided into equal vertical and horizontal units, and then for the controls to be positioned and sized within the control space.

(146) A control definition will typically include initialization data. For example, where a control is a web rendering object, the definition will include initial URLs. When the NIM is opened, the control will navigate to the initial URLs to obtain content and render the NIM. If the control is a GIF, the control could retrieve the GIF file from a content server or from the application server. A NIM definition may optionally include additional tags identifying initialization parameters for different platforms: for example, a URL is suitable for a PC, but a "P-URL" may be provided as well, pointing to content suitable for users viewing NIMs through a personal digital assistant (PDA) or similar device.

(147) Controls are typically installed on the applications server, and may be updated after installation by the applications server. The home NIM code, downloaded when the client becomes "NIM-enabled," includes the then-existing controls. Controls are updated as new controls are installed on the server or when a user requests a NIM that requires a new control. The server may then may download such updates to the client parser application, for example on log-in. The NIM framework allows any control to be positioned and initialized in a control space in a NIM, as discussed above.

(148) An XML specification for control definition and layout, in accordance with one embodiment of the invention, follows:

<CONTROL_LAYOUT> Contains zero or more <CONTROL> tags.

WIDTH="integer" Divides control space into this many evenly spaced columns. Default is

1.

HEIGHT="integer" Divides control space into this many evenly spaced rows. Default is 1.

WIDTHSCALES="TRUE"|"FALSE" Default is true.

HEIGHTSCALES="TRUE"|"FALSE" Divides control space into this many evenly spaced columns. Default is 1.

<CONTROL>

CLASS="string" Class may be "Browser," "GIF reader," or other object for rendering computer readable media.

KIND="string"

ID="string" Must be unique with other controls in this NIM.

LEFT="integer" X position of the control in container unites. Default is 0.

TOP="integer" Y position of the control in container units. Default is 0.

WIDTH="integer" Width in container units. Default is 1.

HEIGHT="integer" Height in container units. Default is 1.

URL="URL" This is read if and only if this control is of class "Browser". This is the URL to which this control navigates. Otherwise, may include address for other control content, e.g. GIF address in applications server.

<CATEGORIES> Contains zero or more <CATEGORY> tags.

<CATEGORY> Adds the NIM to this category. This is the only way categories are specified; i.e. there is no master category list.

NAME="string" This is the name of the category.

(149) A NIM definition may also optionally include home NIM categories 274. A home NIM category used by home NIM 204 is a convenient way for a user to keep track of collected NIMS. When a user adds a NIM to a category 204, the category is added, as a string element, to the categories element 274 of the NIM definition in the user profile. For example, a user may categorize a particular NIM as "entertainment," or "news," or "reference." This category will then be added to the categories element 274 of the MM definition.

(150) A NIM definition may also optionally include an events element 275, which defines actions to certain NIM events. For example, the OnClose event, when a NIM is closed, may be assigned a specific and targeted action, similar to a menu item. An XML specification

for the event element in a NIM definition, in accordance with one embodiment of the invention, follows:

<EVENTS>

<ONCLOSE>

(151) Executes this action list on close.

<ACTION_LIST>

(152) Contains zero or more <ACTION> tags.

<ACTION>

RECIPIENT="address"

MESSAGE="string"

(153) A sample NIM definition, in an XML file format in accordance with the above specification, follows:

<NIM DOMAIN="calculator" KIND="basic">

<FRAME CLASS="Standard" BACKGROUNDCOLOR="#FFFF00"

WIDTH="6" HEIGHT="4" FIXWIDTH="TRUE" FIXHEIGHT="TRUE">

<TITLE TEXT="Basic Calculator" COLOR="#000OFF" JUSTIFY="LEFT"/>

<TITLEBARIMAGE JUSTIFY="LEFT" TILELEFT="1" TILERIGHT="1"

NORMAL=" " INACTIVE=" " HOVER=" " DOWN=" "/>

<BOTTOMBARIMAGE JUSTIFY="LEFT" TILELEFT="1"

TILERIGHT="1" NORMAL=" " INACTIVE=" " HOVER=" " DOWN=" "/>

</FRAME>

<MENU/>

<CONTROL LAYOUT WIDTH="1" HEIGHT="1">

<CONTROL CLASS="Browser" KIND="A" ID="1" TOP="0" LEFT="O" WIDTH="1"

HEIGHT="1"

URL="http://www.dodots.com /dots/Calc/CALCULATOR2.htm"/>

</CONTROL LAYOUT>

</NIM>

(154) The first line of this definition establishes the identification of the NIM definition, as discussed above: it is in the domain "calculator," and the kind of display is "basic."

(155) In one embodiment, the domain will be the domain name associated with the content provider. The domain name is a unique label for the provider or developer of the NIM. The NIM's "kind" is a helpful identifier for locating the NIM, and need not be unique. A NIM may also be identified using a GlobalID, when the NIM is being shared. Since the NIM defined by this XML file is not being shared, it does not have a GlobalID.

(156) The second line of the example XML NIM definition establishes the size and appearance of the NIM frame, defining a NIM viewer in which the NIM content will be placed. The third line ensures that the height and width of the frame are fixed--that is, the size of this frame cannot be adjusted by the user. The fourth and fifth lines establish the title of the NIM--"Basic calculator"--and its location. The next four lines establish the location and placing of the titlebar and bottombar, and relevant images, e.g. mouse-over. Thus, the first part of the example NIM definition defines the NIM frame. The definition of a frame, titlebar, menu and other aspects distinguish a NIM from a browser--the content provider has control over the frame size and every aspect of the NIM's appearance, whereas when a browser is used, the content provider has to adapt to the browser display size, and browser titlebar, menu, logo and other aspects cannot be controlled by the content provider.

(157) The rest of the NIM definition identifies, positions, and initializes the NIM's controls, which, in this case, are contained by the NIM frame. In this example, the next few lines

establish that a single control will start in the upper left corner of the NIM viewer, that the control is of the type "browser," or web-rendering, and that the initialization URL for the control is www.dodots.com /dots/Calc/CALCULATOR2.htm. This URL is typically referred to as the "initialization URL," because it is where the NIM looks for NIM content when it is opened. Where the control is of type "browser," the content will typically be HTML content. However, any standard Internet content--HTML, dHTML, flash, streaming media, or Java, for example--may be used. As discussed earlier, a control, could include types other than a browser. The final two lines of the XML file close the definition file.

(158) It will be appreciated that the NIM is designed such that content consumes the entire frame. In one embodiment, the content for the corners of the frame--the menu and the exit button--and the frame sizing images are served by an application server and referenced when the user logs in. Everything else is developed and served by a separate NIM developer. This differs fundamentally from the current approach to providing web content, in which there is a strong distinction between the viewer application--the browser--and the web page or web content. Using present browser-type technology, the content is trapped within the viewer. To obtain a cohesive application feel and access to application features, the current alternative is to develop custom client applications. NIMs allow a developer to provide an application feel without developing custom client applications.

(159) NIMs and the client parser application have a messaging architecture--the NIM messaging architecture, or NMA--that enables NIMs, controls, and the client parser application to communicate. Messaging, in combination with the NIM definition, gives the content of a NIM access to the application/rendering program--the client parser application--and to other NIMs, allowing true application behavior. The NIM definition,

discussed above, is accessible, flexible, and may be changed by a NIM or a user while the NIM is in use, even after it's been rendered.

(160) The content provider, the user, or other NIMs can change a NIM. For example, the content--which includes the titlebar and menu elements--may be changed by the NIM provider by simply enhanced NIM content, using 20 messaging, so that NIMs can exhibit true application behavior. For example, an online brokerage firm can go beyond providing a NIM that renders stock tracking charts, and allow users to trade on-line via a second NIM that can interact with other NIMs such as the first NIM to help facilitate the activity.

(161) A NIM can be changed by its content, or by another NIM, using messaging. This enables a NIM to, for example, notify a user of events, such as a change in content. A NIM could, for example, remain open, but in a collapsed mode, until a particular event occurs, and could then either expand to normal size, or open another related NIM. For example, if a particular stock hits a predetermined price, the stock tracking chart NIM discussed earlier can notify the user by expanding, by popping up a message, or by opening another NIM (such as a stock trading NIM). Alternatively, the NIM could notify the user of a particular event by coming into focus or changing size or content. These changes could also be made by sending messages from a content or applications server to the NIM.

(162) As illustrated in FIG. 14, all elements of the system can send and receive NMA messages. Message routing between NIMs, from a NIM to itself (that is, between e.g. the frame elements and a control, or one control and another), and from NIMs to the system, are handled by a message interface module 106, which is part of the client parser application 38 in the client 20. In one embodiment, the message interface module 106 resolves addressing queries, executes system-level commands from the NIMs such as

"close all NIMs," and passes messages between NIMs. The message interface 106 may also communicate messages to controls such as the browser class controls, for example "navigate the addressed NIM's browser control to the argument URL." Alternatively, the message interface module 106 may route a message to an application or content server (not shown in FIG. 14) for expanded functionality. In one embodiment, the message interface module 106 uses an HTTP request to access an application program interface (API) call, with data optionally being sent and received in XML format. For example, the message interface 106 could send a message providing user input, such as credit card information to a credit processing module on a web server 82.

(163) In one embodiment, an NMA message has two components: a recipient, or address, and the message body. Both are represented as strings. The address may specify an exact NIM, a kind of NIM or control, a domain, or the system, meaning the overall home NIM display. For example, in one embodiment, the address may be in the form:

#<NTM specifier>:<control specifier>

(164) if the communication is between NIMs in the same domain, or

#<domain>:<NIM specifier>:<control specifier>

(165) or

#<domain>:<NIM specifier>:<NIM id>:<control kind>:<control id>

(166) if the communication is between NIMs in different domains, or

#system

(167) if the communication is to the system.

(168) In one embodiment, if the address is not properly specified it defaults to #system. The message interface 106 in the client 20 can flexibly allow, restrict MM addressing or sending messages--for example, the message interface can ensure that only certain NIMs can send message to a particular NIM. This allows NIM developers to develop coordinated NIMs that can interact, by sending messages to e.g. change content or open one another, without allowing other NIMs to interact with their coordinated NIMs.

(169) In one embodiment, a specifier in n address may be the unique identification of the NIM, control or domain in question: <specifier>:=<ID>. Alternatively, where the kind of NIM or control receiving the message is important but the specific NIM or control is not, the specifier may address a message to the closest matching recipient using a search criteria: <specifier>:=<kind>#<search criteria>. The kind should be a type of NIM or control that is installed in the system. Search criteria may be, for example, "any," "open," or "closed." Finally, where the control is being specified, the specifier may be a symbol, such as ".", indicating that the message is addressed to the sending NIM. For example, in one embodiment, the address #system sends the message to the system. The address #. sends the message to the NIM which sent the message. The address #7 sends the a message to the NIM with the identifier 7. The address #B#open:5 sends a message to the control with the identifier 5, in the first open dot of kind B found in the system.

(170) In one embodiment, certain NIMs will have privileges to send particular messages to certain NIMs, and not to other NIMs. For example, NIMs in a domain may be permitted to send control messages to other NIMs in the same domain, but not to NIMs in other domains. Thus, a NIM provider may have control over messaging between NIMs in his domain, and prevent NIMs in other domains from changing NIMs in his domain. Alternatively, NIM providers may coordinate with NIM providers in other domains, permitting certain messaging privileges between some of the NIMs in their respective domains. The HomeNIM and other system NIMs, such as the login NIM, which are in the system domain, may have certain messaging privileges that no other NIMs have, such as logging the user out or closing all the currently-open NIMs.

(171) In one embodiment of the invention, the second part of the message, the body, is represented as a string of characters. Messages may be sent to the client parser application, to the frame of a NIM, or to a control. Messages may be specific, defined messages, as shown in the following examples, or may be any javascript, which may be sent in or out of NIM content. Examples of defined messages to the client parser application, in one embodiment, are:

Application Message<arg>	Function
Refresh	Refreshes the user's profile.
#have-NIM <NIM-address>	Check if the user has the specified NIM as part of the user's profile.
#delete-NIM <NIM-address>	Remove the specified NIM from the user's profile.
#get-screen-width	Returns the width of the screen.
#get-screen-height	Returns the height of the screen.
#close-all-NIMs	Closes all open NIMs.
#get-NIM-ids <NIM-address>	Returns the NIM ID of the specified NIM.

(172) Messages can also be sent from a NIM to itself, or to another NIM, and the identified actions or functions, specified in the body, are performed on the receiving NIM. The following are examples of messages to a NIM for one embodiment of the invention:

Defined NIM Message<arg>	Function
#set-size<width><height>	Sets the size of the NIM.
#set-width<width>	Sets the width of the NIM.
#set-height<height>	Sets the height of the NIM.
#set-position<x-pos><y-pos>	Sets the position, of screen, of the NIM.
#set-title<title>	Sets the title of the NIM.
#collapse	Collapses, but does not close, the NIM.
#uncollapse	Uncollapses the NIM.
#set-user-sizable<width true/ false><height true/false> sizeable	Establishes whether the NIM is by the user
#set-background-color<color>	Sets background color of the NIM.
#set-title-text-justify<justify-keyword>	

(173) The above examples of NIM messages may, in one embodiment of the invention, be sent to a NIM by another NIM. For example, a user may provide input to a NIM, for example a stock tracking chart NIM, indicating an interest in another NIM, such as a stock trading NIM. The current NIM may then send an "open" message to the second NIM to open it, if it wasn't already open. The current NEM may then send a navigate message (see below) that may include an argument, such as a URL or other content pointer, so that the second NIM could be opened to a specific URL indicated by the first NIM.

(174) Finally, messages may be sent to a control of a NIM, either by the NIM itself, another control, the HomeNim system, or another NIM. As examples, the following control messages are provided in one embodiment of the invention:

Defined Control Message	Function

##<any javascript>	Run any javascript in the control, e.g. javascriptin a browser-type control.
#show	Set visibility control within a NIM.
#hide	Set invisibility of a control within a NIM.
#get-size	Get size of control.
#get-address	Query for unique ID of a control by kind.
#navigate	Navigate the control.

(175) Control messages may be used by a NIM, addressed to its own control, or to the control of another NIM. Control messages may also be sent by the HomeNIM, or by the applications server or content server.

(176) An example of a message to a NIM is window.external.PostMessage("#.:","#collapse)" this is a message from a NIM, to itself, collapsing the NIM. Another example of a message from a NIM to another NIM is "window.external.PostMessage("#mp3#any:","#open")", which is addressed to a NIM with the kind "mp3," but only if it is in the same domain as the sending NIM. The body of the message instructs the message of type mp3 to open. An example of a javascript message is "window.external.PostMessage("#mp3#any:1", "33play( )")", which sends a message to the mp3 NIM control that calls the javascript function play( ).

(177) The operation of the home NIM and individual NIMs has been described. Attention presently turns to different techniques used in accordance with the invention 25 to host NIMs.

(178) As shown and described in relation to FIG. 1, the application server 50 includes a NIM Management Modulel 12, a NIM Templates Database 74, a ShareLink Database 78, and a user profile database 76.

(179) FIG. 15 is a diagrammatic illustration of an embodiment of the NIM Management Modulel 12. NIM Management Modulel 12 may contain the NIM Templates Database 74 and the ShareLink Database 78, discussed in further detail below. NIM Management Module 112 may also contain content 402 for filling in a NIM frame or for rendering Internet pages. Alternatively, content 402 may be stored elsewhere, such as on a Web server similar to the server 82 shown in FIG. 1. As discussed above, the content preferably contains Internet content such as HTML (Hypertext Markup Language), dHTML, and images.

(180) In addition, Management Modulel 12 preferably contains executable procedures 403 for controlling and managing the NIM system. These procedures 403 may include: a Collection procedure 404 for obtaining new NIMs; a Sharing procedure 405 for sharing NIMs with others; Development procedures 406, such as a DevZone procedure 407 and a DevNIM procedure 408, for creating, modifying or deleting NIMs; Publishing procedures 409, such as a PubZone procedure 410 and a PubNIM procedure 411, for publishing NIMs so that they are publically accessible; and Administration procedures 412, such as an AdminZone procedure 413 and an AdminNIM procedure 414, for administering the system. It should be noted that the NIM sharing procedure, discussed in further detail below, may be processed by either the Server module (102 of FIG. 1), or the Sharing procedure (405 of FIG. 15). Control and management of the server and the NIM Management Modulel 12 components will now be discussed in further detail.

(181) FIG. 16 is a diagrammatic illustration of an embodiment of the NIM Templates database 74. In this embodiment, NIM Templates database 74 primarily stores XML NIM definitions in their initial unmodified state as initially designed by a partner and which have

not been altered by a user in any way. The unmodified NIMs are also referred to as "Raw NIMs". The NIM Templatesdatabase 74 is used as the starting point for the development of NIMs which may later be customized or modified by a user, developer, or system administrator, as discussed below.

(182) For ease of explanation, the individual or organization that controls the server (50 of FIG. 1) will hereinafter be referred to as the system provider and the individual or organization who supplies the content will hereinafter be referred to as a partner. It should be understood that a provider, partner, user, developer, and administrator of the system may be distinct entities, the same entity, or a combination of both. Furthermore, as discussed above, each of the above entities is assigned access rights or privileges which permit or forbid that entity from performing different actions on the system.

(183) FIG. 16 illustrates an embodiment of a NIM Template database 74. A NIM identification number (NIM_ID) 416 is stored in a NIM Template table 41.5 in the NIM Templates database 74. A Raw NW, identified by its NIM_ID, includes a plethora of RAW NIM characteristics, including, but not limited to, a Raw NIM creation date 417, which indicates when the NIM was created and is useful when searching for NIMs created during a specific time, a Raw N.I.M definition module 418, and the NIMIndex categories 422 in which the NIM has been categorized. Each NIM when created in typically classified into one or more NIM Index categories by the developer, such as "Applications", "Business", "Entertainment" and "News".

(184) Each NIM is fully configurable and definable. The NIM definition module 418 contains details defining the NIM, such as the look-and-feel 419, of the Raw NIM, initialization URLs (Uniform Resource Locators) 420, and a location 421 of where the developer would

like the NIM to open on a users's computer screen. The look and feel of the NIM is the appearance and function of the NIM interface. The look and feel may characterize the frame or skeleton layout, the graphics used to represent certain functions, such as opening and closing the NIM, whether the frame is sizable, and the appearance and operation of menus in the frame.

(185) The definition module 418 may also contain Initialization URLs 420 which reference resources containing content. The content resources may be HTML (Hypertext Markup Language), dHTML, images, programs such as Java applets, or any other file supported by HTTP. The Initialization URLs 420 contains the name of the protocol required to access the resource, a domain name that identifies a specific computer on a network, such as the Internet, and a hierarchical description of a file location on that specific computer. These files or resources are then used by the home NIM to fill in the frame and controls with content. In addition, the definition module 418 may contain other details such as the location 421 on a user's computer screen where the NIM should initially open.

(186) The NIMIndex may be used to search for, learn about, and collect NIMs. The NIMIndex is typically accessed from either a Web browser, such as Internet Explorer.RTM. or Netscape Navigator.RTM. or from the home NIM, 108 of FIG. 1. A user may search for NIMs by, or according to, any field of the NIM Templates table 415 via the NIMIndex.

(187) A user accessing the NIMIndex from a Web Brow typically navigates to a main NIMIndex web page such as that shown in FIG. 17. FIG. 17 is an illustration of the main NIMIndex Web page 423. A user typically navigates to main NIMIndex Web page 423 from a NIM home-page (not shown), or while anywhere within the NIM Web site by

clicking on the "Collect the DOtS.TM." link 424 in a menu 425. A user may search the NIMIndex by entering a search term in a form 426 and clicking on the "Search" button 428 which implements the Collection Procedure (404 of FIG. 15) to search the NIM Templates database (74 of FIG. 15) for NIMs that match the query. In one embodiment, the NLMIndex may be searched by NIM title, NIM description, or partner, as shown in the pull down menu 430. The user may also browse the NIMIndex by clicking on a link 432 to a NIMIndex category 434 which will navigate the user to a NIMIndex category Web page 440, as shown in FIG. 18.

(188) FIG. 18 is an illustration of a single NIMIndex category, the "Applications" category 435, shown in FIG. 17. A list of sub-categories (not shown), as well as a list of NIMs 442 and their short descriptions are shown in FIG. 18. A user may click on the "more detail" link 444 to be taken to a page displaying a full description (discussed later in relation to FIG. 29B) of the NIM, shown in FIG. 19.

(189) Navigation of the Internet generally occurs through the use of URLs (Uniform Resource Locators), which are the addresses of files or resources accessible on the Internet. The type of resource depends on the Internet application protocol. Using the World Wide Web's protocol, the Hypertext Transfer Protocol (HTTP), the resource can be an HTML (Hypertext Markup Language) page, an image file, a program such as a Java applet, or any other file supported by HTTP. The URL contains the name of the protocol required to access the resource, a domain name that identifies a specific computer on the Internet, and a hierarchical description of a file location on the computer and usually takes the form: "URL=protocol://machine.name[:port]/directory/document.name`?[&argu- ments] " The "protocol" is the Internet protocol used to reach the document or resource. On the Web, the

"protocol" is typically HTTP, but it can take any number of forms, such as ftp (file transfer protocol), file (a local file), gopher (gopher protocol), mailto (electronic mail address), news (Usenet news), telnet and tn3270 (interactive sessions), wais (wide area information servers), or the like.

(190) The "machine.name" is the name of the host come where the document resides (such as www.NIM.com). The ":port" portion of the address is optional and is only necessary when a resource is accessible through a non-standard TCP port number. Although the standard port number for HTTP is 80, there are numerous Web servers on the Internet that use non-standard ports, such as port 8000.

(191) The NIM system, however, may also utilize a proprietary NIM protocol. An example of a URL using the proprietary NIM protocol is: "NIMS:?NIMTemplate=<N/M_ID>"

(192) The NIM protocol URL is used to collect, distribute, and share NIMs. When collecting NIMs the NIM protocol URL is referred to as a NIMLink. When distributing or sharing NIMs the NIM protocol URL is referred to as a ShareLink.

(193) The "NIMs:" term defines the NIM protocol or scheme and is always followed by a colon. The "?NIMTemplate=<NIM_ID" is an argument, where a dollar sign (5) and a question mark (?) are used to denote path and/or search elements. It should be noted that no path is supplied (i.e.: "//path/to/something"). The argument instructs the client parser application (38 of FIG. 1) how to handle a user's selection of a NIM protocol URL and what the NIM protocol URL must do. For example, to obtain a NIM, the argument might read "NIMTemplate=123`, to obtain a Share (discussed below) the argument might read "Share=123", to obtain a Pack (discussed below) the argument might read "Pack=123", etc. The argument can be used to cause the client parser application to do anything within it's

system of functionality by specifying new argument sets to build new types of special client parser application links.

(194) In one embodiment, the address for where the client parser application (38 of FIG. 1) searches the system (10 of FIG. 1) for the NIM Template or ShareLink database (74 and 78 of FIG. 1) is specified within the processed login script or session_config, although it could alternatively be specified within the NIM protocol URL. When a user clicks on a NIM protocol URL (from any where you can place and click on a link, for example in a browser, in a NIM, in email, in a document, etc.), the client parser application processes the NIM protocol URL in the same manner as a browser processes HTTP links and an email program processes mailto links.

(195) When a protocol URL takes the form of a NIMLink, the client parser application responds by obtaining the NIM definition from the NIM Template database, optionally adds the NIM to the user's processed user profile (unless the NIM has been specified to be opened in transient mode, which may be specified in the argument), and optionally opens the NIM on the user's display screen.

(196) In one embodiment, by default, unless specified otherwise, a NIM will be added to a users collection (transient--false) and will be opened (open--true). A NIMLink with arguments may look as follows: "NIMS:?NIMTemplate&transient=true" or "NIMS:?NIMTemplate&open=false". More than one additional argument could be added by appending another argument to the URL which may read as "&argument=value".

(197) A ShareLink (discussed below) is similar to a NIMLink and may read "NIMS:?share=123", where 123 is the SHARE ID referencing the share module within the

ShareLink Database. Pack Links (discussed below) typically read as "NIMS:?pack-123", where 123 is the PACK_ID referencing a pack module within the NIM Template Database.

(198) FIG. 19 is an illustration of a full description of NIM content 446. A graphic of the opened NIM may also be displayed 448.

(199) Once the user decides that he would like to add a NIM to his home NIM, the user clicks on the "get it now" or "Get This Dots.TM. NIMLink 450 (FIGS. 18 and 19) which either runs the Collection procedure (404 of FIG. 15) which obtains that NIM's NIM definition module (418 of FIG. 16) from the NIM Template table (415 of FIG. 16), or opens another Web page as shown in FIG. 20.

(200) FIG. 20 is an illustration of a Web page 452 which might be displayed to the user once the user has clicked on the NIMLink 450. The user is presented with an option of either collecting the NIM 456, or if the user does not have the home NIM application, the user may first download the home NIM by clicking on "Get the homeDotIm" 454. Once the user clicks on the download the NIM button 456, the Collection procedure (404 of FIG. 15) obtains that NIM's NIM definition module (418 of FIG. 16) from the NIM Template table (415 of FIG. 16).

(201) The Collection procedure (404 of FIG. 15) transmits the NIM definition to the user's home NIM, which optionally opens the MM and saves the NIM definition module (418 of FIG. 16) on the user's local processed user profile. All NIM definition modules (418 of FIG. 16) on the user's computer may subsequently be saved to the user profile database, as discussed earlier in this writing. Alternatively, a "preview" button may be provided which transiently displays the NIM on the user's computer screen without adding the NIM to the

user's local processed user profile. The user may also search the NIMIndex from their home NIM.

(202) FIG. 21 is an illustration of the main home NIM graphical user interface (GUI) 464, similar to that shown in FIG. 5. The home NIM displays a list of all NIMs 466 that the user has collected. Furthermore, any NIMs that the user has collected in groups or packs, can be accessed by clicking on the "My Dotpacks'" tab 468. One way to obtain new NIMs is to click on the "Get" button 470, which opens the NIM shown in FIG. 22.

(203) FIG. 22 is an illustration of a get new NIM GUI 474. A list of all NIMs 476 (or a featured subset) that may be collected by the user are displayed. Clicking on the "More Dots'" tab 478 (shown in FIG. 23) displays further NIMs which may be collected.

(204) When a user selects or clicks on any of the NIMLinks 480, NIMLink 480 references the NIM_ID (416 of FIG. 16) for that NIM in the NIM Templates database (74 of FIG. 16). The collection procedure (404 of FIG. 15) receives the NIM_ID (416 of FIG. 16) from the user, locates the NIM definition module (418 of FIG. 16) corresponding to that NIM_ID in the NIM templates database, and transmits the NIM definition module to the user's computer. That NIM may automatically be opened on the user's computer screen. The NIM is saved to the user's list of NIMs on their home NIM (466 of FIG. 21), and the NIM definition module is saved in the user's local processed user profile. Alternatively a "preview" button may be provided which transiently displays the NIM on the user's computer screen without adding the NIM to the user's local processed user profile, as discussed above in relation to the N1MLink. All the NIM definitions that the user has listed on their home NIM are saved to the user profile database either periodically, at a set time, by event, or when the user closes their home NIM. The technique of the invention facilitates a viral distribution architecture.

In other words, the technique of the invention facilitates rampant distribution of generated NIMs, as described below.

(205) Users (or developers) may share NIMs they have collected, and perhaps even modified, with other users (or developers) in accordance with this viral distribution architecture. Because the NIM definition contains basic reference information, such as data to instantiate the NIM and URLs and other references to where the NIM content is located, a NIM is easily and quickly distributed, collected, and shared. By-packaging Internet content and applications as NIMs and referencing the NIMs by NIMLinks, the system advantageously gives Internet content viral characteristics as the NIMs can easily be distributed or shared between users.

(206) Each NIM definition contains just enough information to define and initialize the NIM's components (NIM frame, controls, etc.). For example, this information may contain data to configure the skeleton or frame that is filled in by NIM content from a developer's server. The NIM definition is therefore fairly small in size (-2K), and is therefore easily distributable as an XML file or Blob (binary large object), which is communicated using the same mechanisms (HTTP/HTTPS requests) as regular Web pages.

(207) This is especially useful where a user has collected a NIM or a group of NIMs (Packs) that he would like to send to another user. For example, a user may have an online trading NIM, calculator NIM, and stock research MM all set up in various positions on his screen, and would like to share the entire Pack with a friend who is remotely connected to the Internet.

(208) To share NIMs with others, the system utilizes the Sharelink database 78 of 25 FIG. 15 and the Sharing procedure 405 of FIG. 15. FIG. 24 is a diagrammatic illustration of the

Sharelink database 78. NIM Sharelink database 78 stores a list of all NIMs shared by users, developers, or administrators, in a share table 484. Each NIM or group of NIMs shared is assigned a Share ID 486 which points to a Share module 488. Each Share Module 488 may also include a creation date 490, multiple 30 individual MM definition modules 492, or multiple packs of NIMs that have been shared (Sharepack module 494) containing multiple NIM definition modules 496 and 498.

(209) FIG. 25 is an illustration of a Share NIM's GUI 500. All dots collected by the user (466 of FIG. 21) can be shared with other users by clicking on the "Share" button 502 shown in FIG. 21. Once the user has clicked on the "Share" button 502, the GUI 500 shown in FIG. 25 is launched. The user may then highlight any of the NIMs or packs of NIMs 504 he has collected or created and thereafter share the NIMs or packs of NIMs 504 by clicking on the "Share via email" button 506. It should be noted that other means of distributing the NIMs may be used together with, or instead of, email.

(210) When users share NIMs or NIM packs, their home NIM application generates a 10 share module, which may for example be an XML Blob containing the N1M definition or Sharepack modules shared. The shared NIM XML is then sent to, and saved in, the Sharelink database (78 of FIGS. 1 and 24). The Sharing procedure 405 of FIG. 15 then automatically generates a shared link (ShareLink) that references or points to the address of the shared XML on the Sharelink database. This ShareLink is then sent or 15 distributed (via email or posted on a Web site) to other users.

(211) If a user receives shared NIM(s) or pack(s) and has a home NIM installed on his client computer, then clicking on the Sharelink adds the NIM(s) to the user's home NIM and opens the shared NIM(s) on the user's screen. If a recipient of a shared NIM does not have

the home NIM installed on his computer, then the home NIM is downloaded and installed (with the user's cooperation), the shared MM is added to his local processed user profile, and the NIM is opened.

(212) The NIM management module (112 of FIG. 15) may also be responsible for controlling and managing the development of new NIMs via the DevZone and the DevNIM discussed below.

(213) Because NIM content is based on existing Internet content standards (HTML, DHTML, GIFs, etc.) developers can create MM content using their existing Internet content development tools and methodologies. Therefore, no special hardware or software is required to develop or serve NIM content.

(214) Furthermore, as the application server (50 of FIG. 1) hosts and delivers NIM definitions from the NIM Templates database (74 of FIG. 1) developers merely define and package the NIM content without directly authoring, hosting, or serving the XML NIM definitions. Therefore, no special hardware or software is required on the developer-side to host and serve the NIM content, other than required for their regular Internet content.

(215) Two means are provided for creating NIMs. First, a Developer Zone Web site (DevZone) and second, a set of developing NIMs (DevNIMs). Both means enable NIM developers to create, define, and modify NIM definitions, and to support the NIM development process which results in XML NIM definitions being added to the NIM Templates database and NIMLinks generated.

(216) The DevZone is a Web site where NIM developers can view a list of NIMs they have defined and/or published, add new NIMs, and categorize, view, modify, or delete their existing NIMs. The DevZone is preferably rendered in a Web browser, is hosted on the

Web server (82 of FIG. 1), and is implemented with a DevZone procedure (406 of FIG. 15). To access the DevZone, the developer may typically pass through a secure portal, such as by supplying a login identity and password.

(217) FIG. 26 is an illustration of the main DevZone Web page 510. All NIMs created by the NIM developer appear in a customized NIM list 514 that may only be accessed by that NIM developer. All NIMs created by a developer appear on the NIM list 514, unless they have been deleted by the developer or by a system administrator. The NIM list may contain the NIM name 518, the date the NIM was created 520, and an indication 516 of whether the NIM is in development or accessible by the public in the NIMIndex (i.e. "in-development" or "published").

(218) To access the NIM definition (for modification or review) the developer clicks on a "modify" or "preview" link 524 as transient (e.g. to add the NIM to their home NIM for previewing and testing). By clicking on the "modify" link, the developer is taken to the NIM modification web page, as shown in FIG. 27. Alternatively, by clicking on the "Create a Dot'" button 522, the developer is taken to a web page similar to the NIM modification web page shown in FIG. 27, where the developer may create a new NIM.

(219) FIG. 27 is a partial view of a NIM modification web page 530. To modify an existing NIM, or create a new NIM definition, a developer preferably utilizes Web forms, such as 534 to 542, or any area that contains objects that capture user input, such as text entry spaces, check boxes, and selection buttons. Developers typically fill in forms with information which defines the NIM, where the details might include the NIM's name 534, the URL for any image associated with NIM 540 (as shown in FIG. 18), the URL for a detailed image 542 (448 of FIG. 19), and such details as NIM frame (e.g., size of NIM,

sizeable), layout of the controls (e.g., WebConduit control), and to specify the initial MM content (e.g., the initial target URLs for the WebConduit control, TitleBar, BottomBar), and any categories in which the developer would like the NIM to be listed in the NIMIndex. Once the developer has completed or modified the forms, he may either save or delete the NIM 532. If the developer selects either the development check box 536 or the public check box 538, and then saves the NIM, the DevZone procedure (407 of FIG. 15) generates a XML NIM definition, stores the XML NIM definition in the NIM Templates database (74 of FIGS. 1 and 15) and returns a NIMLink pointing to that NIM which is listed on the NIM list (514 of FIG. 26) on the developer's home NIM. The only difference being that once the developer selects the public check box 538 and saves the NIM, the NIM definition is published utilizing the PubZone publishing procedure (410 of FIG. 15) to a publically accessible portion of the NIM Template Table (415 of FIG. 16), from where users can access, download, and collect the NIM. If the developer selects the development check box 536, the NIM can only be viewed and or modified by the developer and system administrator. It should be noted that the DevZone only allows control of certain characteristics of each NIM. Other characteristics may be set to default while still other characteristics can only be altered by an administrator. In an alternative embodiment, the DevZone may be rendered in a NIM or group of NIMs just as it was rendered in a Web browser. In either embodiment, NIM developers fill out one or more forms specifying NIM definition parameters, an XML NIM definition gets created and stored in the NIM Templates. Database, and a NIMLink gets generated that points to the new NIM. The Developer can then view or debug this NIM by clicking on the NIMLink to add it to his home NIM, or preview as transient, and thereafter render it on his screen. NIM definitions

may also be developed using NIMs and NMA messages. A 30 developer may create Raw NIMs from empty NIM Templates using a development NIM (the DevNIM) on the developer's home NIM.

(220) FIG. 28A to 28D are GUIs of a development NIM (the Dev NIM). A developer may obtain a DevNIM by either collecting the DevNIM in the usual manner, as discussed above, or the system, via the system administrator, may share the NIM with the developer, also as discussed above. The DevNim contains a DevNIM procedure (408 of FIG. 15) which is transmitted to the developer's home NIM, as discussed above.

(221) To create a new NIM, the developer launches the DevNIM and enters a NIM name 550 into the DevNIM. The server then obtains an empty NIM (a NIM with default or no initialization data and with only basic characteristics) from the NIM Template Database using the procedure for collecting NIMs described above, and saves the empty NIM under the supplied new NIM name 550 locally in the developer's processed user profile. The developer may then modify the empty NIM to the required form using the DevNIM. In the preferred embodiment a pull down menu 552 is provided where the developer can select which feature to modify, such as the frame characteristics (FIG. 28B), the titlebar (FIG. 28C), or initialization URLs for different frame or control elements (FIG. 28D).

(222) Each time the developer modifies a setting, the DevNIM, using the DevNIM procedure, sends NMA messages to the newly saved NIM to modify its definition parameters. For example, modifying the NIM's name, size, TitleBars, BottomBars, or WebConduits (as shown in FIGS. 28A to 28C).

(223) Unlike the DevZone, a new XML NIM definition and NIMLink is not generated every time a modification is made. All modifications (during the development cycle) are made

locally to the NIM definition and are stored in the developer's processed user profile. The DevNIM embodiment, therefore, requires a separate publishing step that promotes the newly created NIM definition from the developer's user profile, to the NIM templates database on the application server.

(224) To publish the NIM, the NIM developer categorizes the NIM and the NIM definition is copied from the developer's processed user profile to the NIM Templates database.

(225) A publishing NIM (PubNIM), implemented with Publishing procedures 409, is provided to handle these functions. The PubNIM may therefore be shared or transmitted to the developer along with the DevNIM. The PubNIM contains a PubNIM procedure (411 of FIG. 15) which controls the publication of the NIMs to the NIM template database, as discussed above. The PubNIM procedure sends a NIM definition module to the application server which receives the NIM definition module, extracts the NIM definition from the share module, stores it in the NIM Templates database, and associates the NIM with the developer so that the NIMLink shows up on the developer's NIM list (in their DevZone account).

(226) Alternatively, the new NIM may be published directly from the DevNIM. Once a user is satisfied with the NIM, he may select an option which publishes that NIM definition to the NIM Templates database. It should be noted that a developer may modify his NIMs at any time from the DevNIM.

(227) As mentioned earlier, when a developer is first authorized to create and/or modify NIMs, or at any time thereafter, information about that developer is saved in that developer's user profile (76 of FIG. 1) on the application server.

(228) A developer may also create application programs using NIMs, which a user may access from his client computer. Just as client-side application characteristics (sizing, position, menus) are accessible to content via NMA, the system may offer server-side application functionality, or toolkits, which are accessible through the NMA.

(229) A developer can build a NIM application without implementing, hosting, or supporting complex server or client applications. By using the server toolkits, a developer can develop N1Ms that exhibit server-application behavior by focusing on implementing NIM content (just like standard Internet content).

(230) For example, a NIM's content (an HTML page) may send a message to the system (or server) to request a credit card to be processed. Other toolkit examples may include credit card billing, user profiling, targeted advertising, email, chat rooms, Internet telephony applications, or calendars.

(231) Any server-side application can be made accessible through the NMA, as a toolkit, just as client-side application behaviors are made accessible. In the current implementation, server-side application functions could be offered by a NIM (exposed via javascript functions on a page in a hidden frame). Other NIMs could access this functionality by sending NMA messages to this "Toolkit NIM" calling the functions. A NIM developer may therefore focus on Internet content development while accessing the features, behaviors, and functionality of an application just as if he had developed custom client and server side applications. The NIM management module (112 of FIG. 15) may also be responsible for controlling and managing the administration of the system via the AdminZone and the AdminNIM discussed below.

(232) A system administrator has the power to create, modify or delete users, developers, NIMs, other administrators, or NIMIndex categories, depending on that administrators access privileges. In a similar manner to the DevZone and DevNIM, system administrators may utilize either a Web browser administration zone (AdminZone), or an administration NIM (AdminNIM) which both make use of Administration procedures (412 of FIG. 15).

(233) To access the AdminZone, an administrator typically passes through a secure portal, such as by supplying a login identity and password. Once within the AdminZone, the administrator may search for a NIM by NIM name or title 552, category, developer, developer contact name, or status, as shown in FIG. 29A. The administrator may also selectively search for NIM's in development or publically accessible NIMs 554.

(234) Utilizing an AdminZone procedure (413 of FIG. 15), once the required NIM 20 is located the administrator may modify or delete the NIM in a similar manner to a developer as shown in FIG. 29B, and described above.

(235) Also utilizing the AdminZone procedure, the administrator may manage NIMIndex categories by creating new categories, modifying or deleting existing categories, and/or adjusting the layout of the NIMs within those categories as shown in FIGS. 30A and 30B. For example, an administrator may change a category's name 558, designate the category active or inactive 556, or create sub-categories 560. The system administrator may also select a category or categories for the NIM to appear in, where each NIM may be registered in more than one category.

(236) Finally, utilizing the AdminZone procedure (413 of FIG. 15), an administrator may search for users, providers, or developers and adjust their details, as shown in FIGS. 31A and 31 B. The system administrator may, for example, change a users contact details. In

addition to adding, modifying or deleting NIMs, system administrator may have the task of reviewing NIM submissions from developers and promoting NIMs to the public. A submissions list of newly submitted NIMs may be displayed to an administrator, who may promote the NIM to the public or view the NIM. Once promoted, changes are made to the NIM Templates database and the NIM is automatically removed from the submissions list (again by utilizing the AdminZone procedure (413 of FIG. 15)).

(237) The foregoing discussion has explored the inherent nature of NIMs. Attention now turns to different techniques that may be used to exploit information that is associated with the use of NIMs. In particular, the following discussion is directed toward the accumulation of statistical information that is only available in view of the architecture of the present invention.

(238) Currently, the predominant method of tracking and collecting user online behavior is severely limited for a number of reasons. First, most Internet use or visitor statistics are single-dimensional (linear, sequential) because Internet content is presented to users one full-screen page at a time. Second, users visit and leave sites so rapidly their visits are barely meaningful. Third, user's browsing habits are often discontinuous (browsers give users navigational bypass controls--back, forward, home, refresh, stop, etc.). Fourth, user behavior tracking is limited from a single site's server point of view. Current use statistics are plagued with the challenge of tracking continuous user behavior (especially from a cross-company perspective), with more than a single dimension of use context. Finally, because a computer may have multiple users, or a single user may use multiple computers, tracking continuous user on-line behavior is extremely difficult.

(239) One of the advantages of the NIM system as illustrated in FIG. 1, is that the Server 50 is able to track continuous, long-term NIM use information about each user. This is because the NIM server, through communication with the home NIM, can track each NIM event performed by each user. Therefore, it is possible to track each individual user's entire NIM use activity from the moment the user downloads the home NIM.

(240) Referring to FIG. 32, in one embodiment of the invention the following events may be tracked by the Event Log Module 98 (within the client computer 20 of FIG. 1): home NIM Download Event 704 NIM Download Event 706 NIM Display Event 708 Web Click-Through Event 710 Page-View Event 712 First NIM Installation Event 714 First home NIM Startup Event 716 Transient Mode Event 718 Share NIM Received Event 720 NIM Pack Received Event 722.

(241) A home NIM Download Event 704 is logged when the user clicks on a link to request the home NIM user application. Preferably, the start time 752, and the end time 754 are recorded for this event. Also recorded is the provider ID 750 which is a parameter (generally, an integer) that represents the content provider partner who provided the link to the user.

(242) NIM Download Event 706 is logged when the home NIM acquires a NIM via a NIMLink. The start time 752, the end time 754, and the provider ID 750 are recorded for this event. Also recorded is the NIM ID 756 which is a parameter (generally, an integer) that represents the NIM that was just downloaded.

(243) NIM Display Event 708 is logged when a user activates a NIM. The NIM ID 756, the start time 752, and the end time 754 are recorded for this event. Web Click-Through Event 710 is logged whenever a user links from a NIM to a full-screen browser. This can occur

when a user clicks on a link in the NIM, or it can occur automatically through the NIM messaging, or directly through the content provider. The NEM ID 756, the start time 752, and the Internet address 758 of the link are recorded. Page-View Event 712 is logged whenever a user views a page of content within a NIM. The NIM ID 756, and the start time 752 are recorded for this event.

(244) First NIM Installation Event 714 is logged the first, a NIM or NIM Pack is installed from a web site. This event is logged only once for each user account. The NIM ID 756, start time 752, and end time 754 are recorded for this event.

(245) First home NIM Startup Event 716 is logged when the home NIM runs for the first time. This event is logged only once for each user account. The start time 752 is recorded for this event.

(246) Transient Mode Event 718 is logged when the home NIM runs in transient mode. Transient mode occurs when the home NIM runs before the user has logged in. The start time 752, and the end time 754 are recorded for this event.

(247) Shared NIM Received Event 720 is logged for each NIM a user receives as part of a share. If a NIM Pack is shared, this event will be recorded for each NIM in the shared pack. The NIM ID 756, the start time 752, and the end time 754 are recorded for this event.

(248) Shared NIM Pack Received Event 722 is logged for each NIM Pack a user receives as part of a share. Thus, when a NIM Pack is shared, an Event 720 will be logged for each NIM in the NIM Pack, while an Event 722 will be logged once for the NIM Pack itself. The start time 752, and the end time 754 are recorded for this event.

(249) The events listed above are tracked in one particular embodiment. Other embodiments may track more or perhaps fewer events. This comprehensive event tracking is possible

because each user event can be identified by the NIM Server through communication with the home NIM. Additional events may include tracking when a user sends a share or tracking when a user sends a NIM or a NIM Pack.

(250) FIG. 33 shows a typical series of user actions 800 as they are tracked by the Event Log Module 98. First, a user may request to download the home NIM application (step 802) from either a partner's web site or the NIM Server 50. The Event Log Module 98 records a home NIM Download Event, as shown with field 704 of FIG. 32. The start time 752, and the end time 754 are preferably recorded. Also, the provider ID 750 of the site from where the home NIM download request was received is recorded.

(251) Returning to FIG. 33, the user subsequently activates the home NIM for the first time (step 804). The Event Log Module 98 records a First home NIM Startup Event 716, as shown in FIG. 32. The start time 752 is preferably recorded. In addition, the home NIM is activated and the user lot yet logged in, a Transient Mode Event 718 is logged and the start time 752 is recorded.

(252) As shown in step 806 of FIG. 33, the user logs into the home NIM. When this occurs, the end time 754 may be recorded for the Transient Mode Event 718.

(253) A user download of a new NIM (step 808) may be from a partner's web site or the NIM Server. When this occurs, the raw NIM definition is copied into the user's User Profile 76. The event log 98 records two events. First, because this is the first NIM the user has installed, a First NIM Installation Event 714 is recorded. The start time 752, the end time 754, and the provider ID 750 of the download site are preferably recorded. The second event recorded is a NIM Download Event 706. The Event Log Module 98 preferably tracks the NIM ID 756, the provider ID 750, the start time 752, and the end time 754 for this event.

The next thing a user may do is open the NIM (step 810). This consists of retrieving the NIM definition from the user's User Profile and getting NIM content from the provider 82, as discussed above. The NIM is displayed for the user and the Event Log Module 98 records a NIM Display Event 708. However, at this point, the Event Log Module 98 can only record the start time 752, and the NIM ID 756 for this event. The end time 754 is recorded when the NIM is closed.

(254) For every page of content a user views within a NIM 812, a Page-View Event 712 is recorded. Some page views may require content from the provider 82. The NIM ID 756, and the start time 752 are recorded for this event.

(255) The NIM may also enable the user to click on a link that results in navigating to a full screen web browser (step 814). When a user does this, a Web Click-Through Event 710 is recorded. The Event Log Module 98 records the NIM ID 756, the start time 752, and the URL of the web site that is passed from the NIM content to the browser 758.

(256) When the NIM closes (step 816), the end time 754 for the NIM Display Event 708 is recorded. When the user logs out of the home NIM (step 818), the event log is uploaded to the Server 50 (of FIG. 1).

(257) In one embodiment of the invention, the previously described Event Log Module 98 (within the client computer 20 of FIG. 1) tracks user events in the home NIM user application and uploads the information to the Statistics Database 80 (of the server computer 50 of FIG. 1) at predetermined intervals alternate embodiments, the Event Log 700 (in FIG. 32) may be processed by the NIM Server before it is stored in the Statistics Database 80. For example, the NIM Server may process NIM use status information for each user that is currently logged in.

(258) The Statistics Database 80, illustrated in FIG. 34, preferably lists every event 1002 by every user of home NIMs along with the corresponding fields associated with each event. For-example, if a NIM Display Event is recorded, the User ID 1004 of the user that performed the event is listed, the start time 1006 is listed, the end time 1008 is listed, and the NIM ID 1010 is listed. If a Web Click-Through Event is recorded, the User ID 1004 is listed, the NIM ID 1010 is listed, the start time 1006 is listed, and the URL of the web site 1014 is listed. The Statistics Database 80 therefore allows the list of events to be easily referenced and searched by each event or by each of the fields associated with the events.

(259) Referring to FIG. 35, the Statistical Analysis Module 900 uses the Statistics Database 80 in order to provide various services for the content provider partners 82. Preferably, the Statistical Analysis Module 900 includes a Multi-Dimensional Consumer Profile Module 902, a Real-Time Advertising Module 904, and a Pack Building Module 906, as discussed below.

(260) A primary advantage of the present invention is that, because NIMs are used in groups and are used more often and for longer periods of time than web pages or web sites, real-time multi-dimensional NIM use data (that's a function of which NIMs are activated simultaneously) can be accumulated. In accordance with an embodiment of the invention, this accumulated data is used to generate a multi-dimensional consumer profiling database. The Multi-Dimensional Consumer Profile Module 902 uses information from the Statistics Database 80 to examine, for each user, the start time, and the end time of each NIM Display Event It then determines the NIMs (using the NIM IDs) that are opened simultaneously for each user. The Module 902 determines, for every selected NIM, the other NIMs that a given user may use in conjunction with the selected NIM. The Module 902 also determines how

often these other NIMs are used simultaneously with the selected NIM. For example, Company X provides a NIM for selling its books. The Multi-Dimensional Consumer Profile Module 902 determines for Company X that a particular user has a NIM related to finance activated 30% of the time the user has the book-selling NIM acted, a NIM related to computers 20% of the time the user has the book-selling NIM activated, and a NIM related to wedding gifts 5% of the time the user has the book-selling NIM activated. This will provide Company X with a more complete profile of the user's interests.

(261) The Real-Time Advertising Module 904 determines the NIMs that each user has displayed at any given moment. This information is used by a content provider partner or by the NIM Server to target advertising information. For example, if a user has a NIM related to sports displayed simultaneously with Company X's book-selling NIM, Company X uses this information to stream an advertisement for a sports book. In one embodiment, this is accomplished by associating each NIM with a context keyword. This is done by incorporating the context keyword into the NIM definition or, alternatively, by maintaining a table of NIMs and their corresponding context keywords. For example, the NIM related to sports is associated with the context keyword "sports." Moreover, the Real-Time Advertising Module 904 may combine the real-time user information with the historical user information from the Statistics Database 80 to provide advertisers with a complete picture of a user's interests.

(262) The Pack Building Module 906 uses the Statistics Database 80 to determine which NIMs are being used simultaneously. The Module 906 also determines which NIMs are being shared as NIM Packs. From this, the Module 906 provides information to content provider partners about which NIMs should be bundled together. In alternate embodiments,

the Module 906 builds a NIM Pack based upon the information it processes. For example, if the Pack Building Module 906 determines that an airline NIM is being used with a hotel NIM and a car rental NIM, the Module 906 may build a NIM Pack with a restaurant NIM.

(263) Additionally, in one embodiment of the present invention, the NIM Server 82 may track the content within a NIM in a Content Database 1050, as illustrated in FIG. 36. A content descriptor 1052 which may be a string describing the content that is shown within the NIM is recorded for content shown in the NIM. For example, if a NIM displayed an advertisement for an automobile followed by an advertisement for a restaurant, the two recorded content descriptors might say "automobile ad" and "restaurant ad." In addition, the NIM ID 1054, the start time at which the content is displayed 1056, and the end time 1058 are all preferably recorded for each content descriptor.

(264) Referring to FIG. 35, The Content Analysis Module 950 is able to correlate, at any moment, the content displayed to the user as recorded in the Content Database with the user's NIM activity recorded in the Statistics Database. For example, if one NIM displays to a user an advertisement for a travel book, the user may open a NIM related to Florida, a NIM owned by a specific airline, and a NIM owned by a car rental company. This pattern of user behavior will allow the company that provides the travel book advertisement to better understand the effect of the advertisement on the user. The company may use this information to make cross-promotions with other NIM providers, or, simply to provide more effective targeted advertisements.

(265) In an alternative embodiment, each of the content providers may track its own content information. The content providers could then compare its content information with the user information provided by the Statistics Database of the NIMServer.

(266) Finally, referring to FIG. 37, all of the user event information may be used in conjunction with user information provided at login. During the login process, the user may be required to enter demographic information such as age, marital status, etc. In one embodiment, this information is stored in a User Account Database 1100. Each User ID 1102 is listed along with the corresponding user information 1104. Therefore, it is possible to match the user events with personal information about the particular user to give advertisers or NIM content providers a more complete behavior profile of each user.

(267) The foregoing description, for purposes of explanation, used specific nomenclature to provide a thorough understanding of the invention. However, it will be apparent to one skilled in the art that the specific details are not required in order to practice the invention. In other instances, well known circuits and devices are shown in block diagram form in order to avoid unnecessary distraction from the underlying invention. Thus, the foregoing descriptions of specific embodiments of the present invention are presented for purposes of illustration and description. They are not intended to be exhaustive or to limit the invention to the precise forms disclosed, obviously many modifications and variations are possible in view of the above teachings. The embodiments were chosen and described in order to best explain the principles of the invention and its practical applications, to thereby enable others skilled in the art to best utilize the invention and various embodiments with various modifications as are suited to the particular use contemplated. It is intended that the scope of the invention be defined by the following claims and their equivalents.

SUBSTITUTE SPECIFICATION (MARKED UP)

DISPLAYING TIME-VARYING INTERNET BASED DATA USING APPLICATION MEDIA PACKAGES

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] The present application is a continuation of and incorporates by reference U.S. Non-Provisional patent application Ser. No. 09/558,925, filed Apr. 26, 2000, now U.S. Patent No. 7,660,868, which claims priority from and incorporates by reference U.S. Provisional Application ser. nos. 60/131,083, filed Apr. 26, 1999, 60/131,114, filed Apr. 26, 1999, 60/131,115, filed Apr. 26, 1999, 60/176,687, filed Jan. 18, 2000, and 60/176,699, filed Jan. 18, 2000. The present application claims priority to U.S. Non-Provisional patent application ser. no. 09/558,925, filed Apr. 26, 2000, now U.S. Patent No. 7,660,868, and each of the aforementioned applications to which it claims priority.

[0002] The present application is also related to and incorporates by reference the following U.S. patent applications: Non-Provisional Application ser. no. 09/558,922, filed April 26, 2000, now U.S. Patent No. 7,756,967; Non-Provisional Application ser. no. 09/558,923, filed April 26, 2000; Non-Provisional Application ser. no. 09/558,924, filed April 26, 2000, now U.S. Patent No. 7,356,569; Non-Provisional Application Ser. No. xx/xxx,xxx, filed October 31, 2007 titled "Component For Accessing And Displaying Internet Content" (attorney docket no. IMS 07-05-01); Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,286, filed Oct. 31, 2007, titled "Server Including Components For Accessing And Displaying Internet Content And For Providing Same To A Client" ~~(attorney docket no. IMS~~

~~07-05-02)~~; Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,392, filed Oct. 31, 2007, titled "Method For Accessing And Displaying Internet Content" ~~(attorney docket no. IMS 07-05-03)~~; Non-Provisional application Ser. No. ~~xx/xxx,xxx~~ 11/932,427, filed Oct. 31, 2007, titled "Component For Coordinating The Accessing And Rendering Of An Application Media Package" ~~(attorney docket no. IMS 07-05-04)~~; Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,456, filed Oct. 31, 2007, titled "Tracking and Tracing User Activity with Application Media Packages" ~~(attorney docket no. IMS 07-05-05)~~; Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,585, filed Oct. 31, 2007, titled "System and Methods for Creating and Authoring Internet Content using Application Media Packages" ~~(attorney docket no. IMS 07-05-07)~~; Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,630, filed Oct. 31, 2007, titled "Methods of Obtaining Application Media Packages" ~~(attorney docket no. IMS 07-05-08)~~; Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,663, filed Oct. 31, 2007, titled "Indexing, Sorting, and Categorizing Dots" ~~(attorney docket no. IMS 07-05-09)~~; Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,692, filed Oct. 31, 2007, titled "System and Methods of Messaging between Application Media Packages" ~~(attorney docket no. IMS 07-05-10)~~; and, Non-Provisional Application ser. no. ~~xx/xxx,xxx~~ 11/932,763, filed Oct. 31, 2007, titled "Component For Accessing And Displaying Internet Content In Association With a Web Browser Application" ~~(attorney docket no. IMS 07-05-11)~~.

~~COPYRIGHT NOTICE~~

~~**[0003]** A portion of the disclosure of this patent document contains material which is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure as it appears in the~~

Patent and Trademark Office patent file or records, but otherwise reserves all copyright rights whatsoever.

BACKGROUND OF THE INVENTION

1. Field of the Invention.

[0004] The present invention is related to accessing and viewing Internet content, and more specifically to a method and apparatus for providing a unique frame, independent of a Web Browser application and window, for the retrieval and display of such content.

2. Description of the Prior Art.

[0005] A user operating a client computer typically accesses the Internet by using a viewer application, such as a browser to view Internet content provided at a destination address, typically a web page. In this context, Internet content and web applications are designed to fill the entire web page. It is known to divide the Internet content into different regions of a single web page. For example, personalized web pages can be specified, such that a user views a variety of content sources in a single page, such as stock information, weather information, and sports information, which is aggregated at the server that delivers the web page to the user, who then views the aggregated content in a single web page. Observe that even when disparate content is aggregated, in this manner, it is reassembled into a full web page and is served through a full-screen browser.

~~**[0006]** Users and application developers therefore have limited control over the presentation of internet content: content is typically trapped within the frame of the browser. A developer's only alternative to engaging a user page-by-page in a browser is to develop, distribute, and support custom client software. In the Web browser scenario, it is the content provider, not the user that aggregates the information that is viewed by the user. Thus, the user is not in a position to separately aggregate the content at a client computer, instead the user is constrained to view the content that has been delivered in the manner provided by the server computer hosting the web page. There is a growing desire for individual users to fully control the aggregation and presentation of content and web applications that appears on a client computer.~~

~~**[0007]** A user who wishes to view multiple web pages or applications can open multiple instances of a browser. However, the user will not be able to view each "full-screen" page at the same time. Instead, the user must adjust the windows corresponding to each browser instance and view only part of each page. The information appearing in each browser is not designed for viewing in this manner. Thus, the user cannot create an optimized display of content from multiple sources.~~

~~**[0008]** Currently, content providers and end users have limited tools to alter the browser in which content appears. That is, the controls associated with a browser are not fully configurable. Thus, the vendor of a browser is in a position to brand the browser and regulate the controls associated with the browser. There is a growing desire for content providers to not only fill a browser with their content, but to also fully brand and control the frame in which the content appears. Further, in some instances, content providers desire to limit the controls~~

associated with a browser or viewer, so that a user is more inclined to view a single set of content, for example, by having limited access to previously viewed content. However, the current browser applications provide very limited control to a user or content provide to alter the frame and controls provided by the browser window.

[0009] In summary, therefore, the current model of the Internet has the following attributes and limitations:

a) Internet content is typically viewed one page at a time, with each page displayed serially displayed in a browser application window which typically takes up the majority of the user's computer screen real estate.

b) Internet content is designed for delivery in web pages. Even if content is modularized, it is reassembled into a full web page and viewed serially in the window of browser application taking up a significant portion of the user's desktop.

c) There is a distinction both visually and architecturally between the "viewer application" (browser) and the "content/document" (web page) such that a browser window is not tailored to the content being displayed, but rather is capable of displaying any web content.

d) Internet content is effectively limited and trapped within the "frame" of the browser (viewer application). Therefore, content developers, users

and web-application developers are limited in how the user experience is controlled.

e) Although internet programming technologies (such as Java script, CSS, layers, flash, etc.) are giving web pages more functionality, the pages have limited access to application functionality such as access and control of the window and frame, the size of the frame, branding, application behavior such as size and menu items, etc.

f) "Web-applications" such as web-mail and web calendars are being packaged and viewed through the page-by-page web model. Even though web-applications are being implemented by many online companies, the web is currently a destination page-based model where, for example, a user visits one page, then goes to another page and so on. It is therefore a sequential, linear experience, e.g., one full page at a time.

[0010] In view of the foregoing, there is a need in the art to provide a technique for accessing multiple instances of distributable computer readable web content in which these instances are typically smaller than the full pages used in current web pages and web applications, and which may be displayed in user- or content provider-controlled frames. Preferably, such techniques allow such access to be done simultaneously. There is a further need for providing the user with flexibility in selecting, collecting, relating and viewing such web content and for giving the content provider flexibility in directing media to a specific user and controlling the framework in which media, such as

web content, is presented. Finally, there is a need to gather more accurate information regarding the type of content that a user enjoys, so that the user can be automatically provided with this content.

SUMMARY OF THE INVENTION

[0011] Accordingly, the present invention is directed to systems and methods by which predefined web content or predefined references to web content may be accessed and displayed on a web-connected client computer. The method relies on the structure defined herein as an Application Media Package. Application Media Packages are web browser-readable code that is executed on a non-browser-based installed client application. The client application, referred to herein as an Application Media Viewer, executes independently from a web browser. The Application Media Viewer parses and executes the Application Media Package code to create the user experience. The Application Media Viewer is capable of loading and simultaneously running a plurality of different Application Media Packages.

[0012] Application Media Packages are programmed to access and display web data including media content. By design, Application Media Packages may graphically reflect the type of data as well as time varying changes in the data. Unlike web pages which have formatting restrictions that are dictated by the browsers into which they are loaded, Application Media Packages have a graphically definable format which is the graphical user interface (GUI) as presented on a user's client computer. Furthermore, unlike web pages together with

~~browsers, Application Media Packages are not generally navigable. They are typically constructed to represent specific data for a specific application.~~

~~**[0013]** Each Application Media Package contains both code and presentation display data or network reference to that data. Together with the Application Media Viewer, the program combination is capable of accessing remote server data that may be processed and/or displayed, modifying on-screen appearance or functionality according to user events, processing web events, client computer events, etc.~~

~~**[0014]** The Application Media Package may operate in a windowed operating system and uniquely operate without the use of a generalized application window. Therefore the Application Media Package graphical user interface is wholly definable by the Application Media Package software developer. It may therefore be designed to exclude the window display requirements associated with the conventional window based operating system applications.~~

~~**[0015]** Application Media Packages may be coded independently of the Application Media Viewer client platform. Thus, independent third party developers may create and post Application Media Packages according to guidelines provided by the creator of the Application Media Viewer. As Web users are increasingly reluctant to download applications from the Web and onto their computers, the Application Media Viewer provides a single trusted application that is downloaded and installed one time.~~

[0016] The Application Media Viewer then provides a layer of security for Web users so that they may download Application Media Packages without the same exposure to compromising their computer as with installed applications. This security is provided by virtue of a trusted module platform together with the module's web browser readable code. This is in contrast to a typical client application which is directly machine executable and may have broader access privileges to the client computer's resources.

[0017] According to an aspect of the invention a physical memory device is provided having stored thereon a first computer software component comprising: a definition of a graphical user interface within which content in a web browser readable language may be presented; an element for causing said user interface to be rendered outside of and without utilization of a web browser application; an element for causing said content to be presented in said graphical user interface; and a network reference to said content for presentation within said user interface.

[0018] The above is a summary of a number of the unique aspects, features, and advantages of the present invention. However, this summary is not exhaustive. Thus, these and other aspects, features, and advantages of the present invention will become more apparent from the following detailed description and the appended drawings, when considered in light of the claims provided herein.

~~**BRIEF DESCRIPTION OF THE DRAWINGS**~~

~~**[0019]** In the drawings appended hereto like reference numerals denote like elements between the various drawings. While illustrative, the drawings are not drawn to scale. In the drawings:~~

~~**[0020]** Fig. 1 is an illustration of the overall architecture of a network in which the present invention may operate.~~

~~**[0021]** Fig. 2 is an illustration of the elements of an Application Media Package (Dot) according to one embodiment of the present invention.~~

~~**[0022]** Fig. 3 is an illustration of the elements of an Application Media Package (Dot) definition according to one embodiment of the present invention.~~

~~**[0023]** Fig. 4 is an illustration of an instantiation of a generic GUI according to one embodiment of the present invention.~~

~~**[0024]** Fig. 5 is an example of an instance of an Application Media Package (Dot) according to one embodiment of the present invention.~~

~~**[0025]** Fig. 6 is an illustration of the communication flow associated with the acquisition and instantiation of an Application Media Package (Dot) according to one embodiment of the present invention.~~

~~**[0026]** Fig. 7 is all illustration of the communication pathways associated with the building of a user profile, and the acquisition of Dots through links, packs, and sharelinks according to one embodiment of the present invention.~~

~~**[0027]** Fig. 8 is an illustration of the communication pathways associated with the acquisition and instantiation of an Application Media Package (Dot) according to one embodiment of the present invention.~~

~~**[0028]** Fig. 9 is an example of a display showing Application Media Packages (Dot) available for downloading as well as Application Media Packages (Dots) already downloaded according to one embodiment of the present invention.~~

~~**[0029]** Fig. 10 is an illustration of the communication pathways associated with the instantiation of an already-acquired Application Media Package (Dot) according to one embodiment of the present invention.~~

~~**[0030]** Fig. 11 is a flow chart illustrating the download process of an Application Media Package (Dot) and Application Media Viewer (Home Dot) according to an embodiment of the present invention.~~

~~**[0031]** Figs 12A, 12B, and 12C are illustrations of Application Media Packages arranged in groups, and arranged in vertical and horizontal Blocks (position-justified groups), respectively, according to an embodiment of the present invention.~~