8. The apparatus of claim 7, wherein the processor is further configured to execute programmed instructions stored in the memory further comprising providing a sending device session content web page to the sending device, the session content web page configured to, when executed by the sending device, generate and send a request for a sending device synchronization web page, the request for the sending device synchronization web page including the synchronization identifier.

9. The apparatus of claim 7, wherein the processor is further configured to execute programmed instructions stored in the memory further comprising providing a sending device synchronization web page to the sending device, the sending device synchronization web page configured to, when executed by the sending device, send the request to establish the first web socket connection and the one or more cookies and redirect URL over the first web socket connection.

10. The apparatus of claim 9, wherein the processor is further configured to execute programmed instructions stored in the memory further comprising providing a receiving device synchronization web page to the receiving device, the receiving device synchronization web page configured to, when executed by the receiving device, generate the request to establish the second web socket connection, receive the one or more cookies and the redirect URL, and redirect a web browser of the receiving device based on the redirect URL.

11. The apparatus of claim 10, wherein the sending device synchronization web page and the receiving device web page are the same synchronization web page and the synchronization web page is configured to, when executed by the sending device or the receiving device, determine a behavior based on whether another cookie including the synchronization identifier is included in a request for the synchronization web page or a response including the synchronization web page.

12. The apparatus of claim 7, wherein the receiving device session content web page is a same version of the sending device session content web page or a different version of the sending device session content web page adapted for a type of the receiving device and served from a different location than sending device session content web page.

13. A non-transitory computer readable medium having stored thereon instructions for synchronizing web sessions comprising machine executable code which when executed by a processor, causes the processor to perform steps comprising:

receiving a request to establish a first web socket connection with a sending device, the request to establish the first web socket connection, and establishing the first web socket connection in response to the request to establish the first web socket connection;

receiving a request to establish a second web socket connection from a receiving device, the request to establish the second web socket connection including the synchronization identifier, establishing the second web socket connection in response to the request to establish the second web socket connection, and notifying the sending device when the second web socket connection is established;

receiving one or more cookies including session information and a redirect uniform resource locator (URL) from the sending device in response to the notification and over the first web socket connection; and

forwarding the one or more cookies and the redirect URL to the receiving device over the second web socket connection, wherein the redirect URL is associated with a web page that, when executed by the receiving device, is configured to comprise the session information.

14. The medium of claim 13, wherein the machine executable code when executed by the processor further causes the processor to perform steps further comprising providing a sending device session content web page to the sending device, the session content web page configured to, when executed by the sending device, generate and send a request for a sending device synchronization web page, the request for the sending device synchronization web page including the synchronization identifier.

15. The medium of claim 13, wherein the machine executable code when executed by the processor further causes the processor to perform steps further comprising providing a sending device synchronization web page to the sending device, the sending device synchronization web page configured to, when executed by the sending device, send the request to establish the first web socket connection and the one or more cookies and redirect URL over the first web socket connection.

16. The medium of claim 15, wherein the machine executable code when executed by the processor further causes the processor to perform steps further comprising providing a receiving device synchronization web page to the receiving device, the receiving device synchronization web page configured to, when executed by the receiving device, generate the request to establish the second web socket connection, receive the one or more cookies and the redirect URL, and redirect a web browser of the receiving device based on the redirect URL.

17. The medium of claim 16, wherein the sending device synchronization web page and the receiving device web page are the same synchronization web page and the synchronization web page is configured to, when executed by the sending device or the receiving device, determine a behavior based on whether a cookie including the synchronization identifier is included in a request for the synchronization web page or a response including the synchronization web page.

18. The medium of claim 13, wherein the receiving device session content web page is a same version of the sending device session content web page or a different version of the sending device session content web page adapted for a type of the receiving device and served from a different location than sending device session content web page.

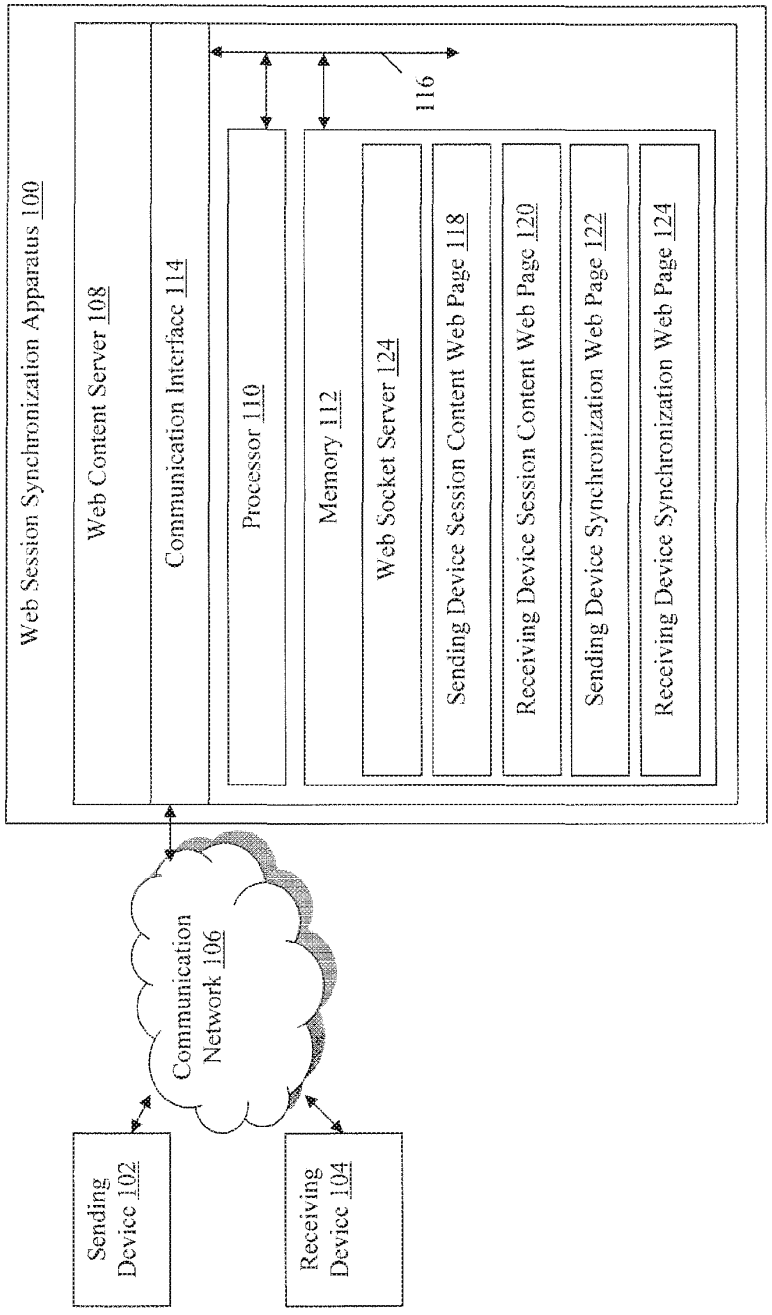

Web Session Synchronization Apparatus 100
Web Content Server 108
Communication Interface 114
Processor 110
Memory 112
116
Web Socket Server 124
Sending Device Session Content Web Page 118
Receiving Device Session Content Web Page 120
Sending Device Synchronization Web Page 122
Receiving Device Synchronization Web Page 124
Communication Network 106
Sending Device 102
Receiving Device 104

FIG. 1

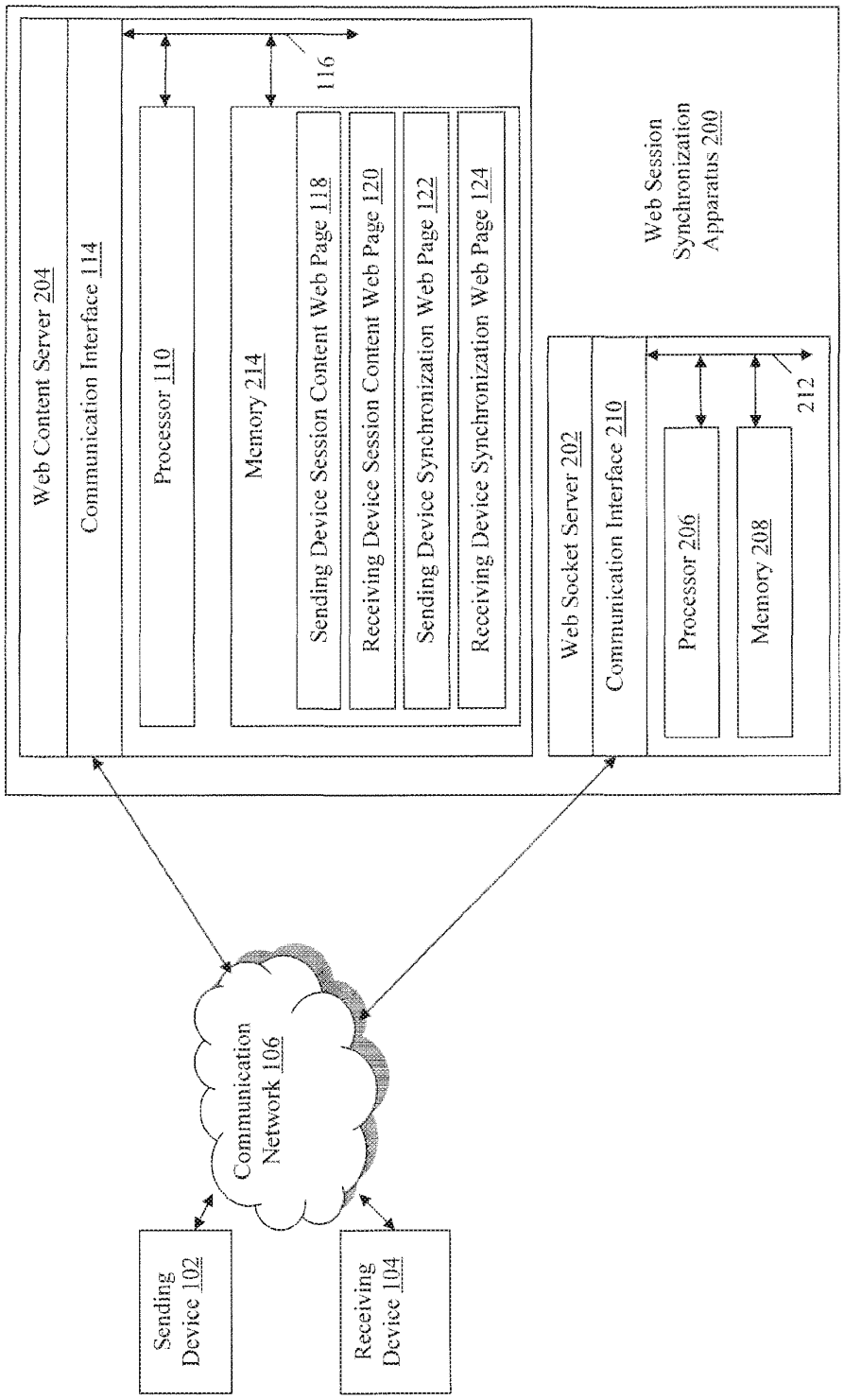

Web Content Server 204
Communication Interface 114
Processor 110
Memory 214
116
Sending Device Session Content Web Page 118
Receiving Device Session Content Web Page 120
Sending Device Synchronization Web Page 122
Receiving Device Synchronization Web Page 124
Web Session Synchronization Apparatus 200
Web Socket Server 202
Communication Interface 210
Processor 206
Memory 208
212
Communication Network 106
Sending Device 102
Receiving Device 104

FIG. 2

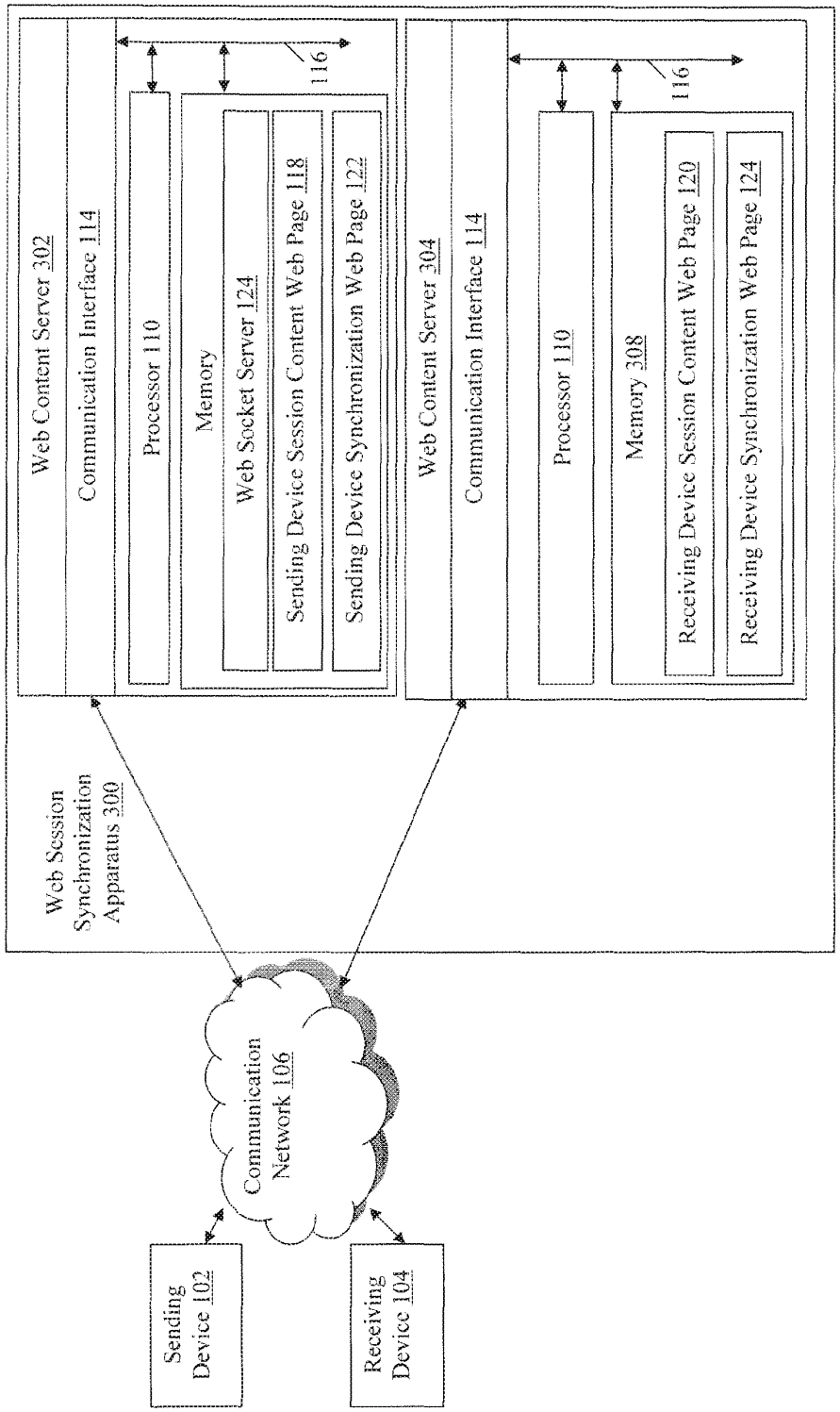

Web Session Synchronization Apparatus 300
Web Content Server 302
Communication Interface 114
Processor 110
Memory
Web Socket Server 124
Sending Device Session Content Web Page 118
Sending Device Synchronization Web Page 122
116
Web Content Server 304
Communication Interface 114
Processor 110
Memory 308
Receiving Device Session Content Web Page 120
Receiving Device Synchronization Web Page 124
116
Communication Network 106
Sending Device 102
Receiving Device 104

FIG. 3

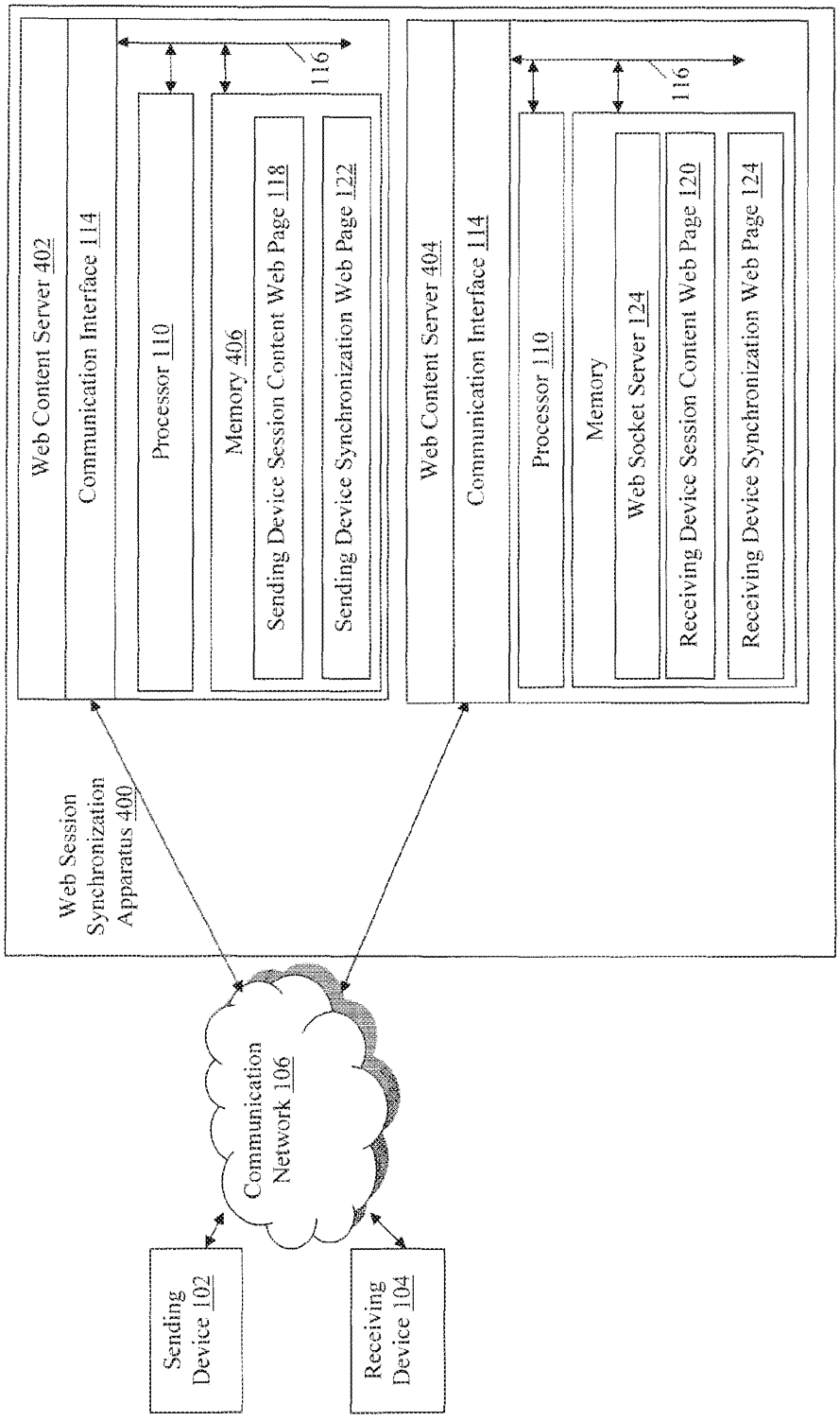

Web Session Synchronization Apparatus 400
Web Content Server 402
Communication Interface 114
Processor 110
Memory 406
Sending Device Session Content Web Page 118
Sending Device Synchronization Web Page 122
116
Web Content Server 404
Communication Interface 114
Processor 110
Memory
Web Socket Server 124
Receiving Device Session Content Web Page 120
Receiving Device Synchronization Web Page 124
116
Communication Network 106
Sending Device 102
Receiving Device 104

FIG. 4

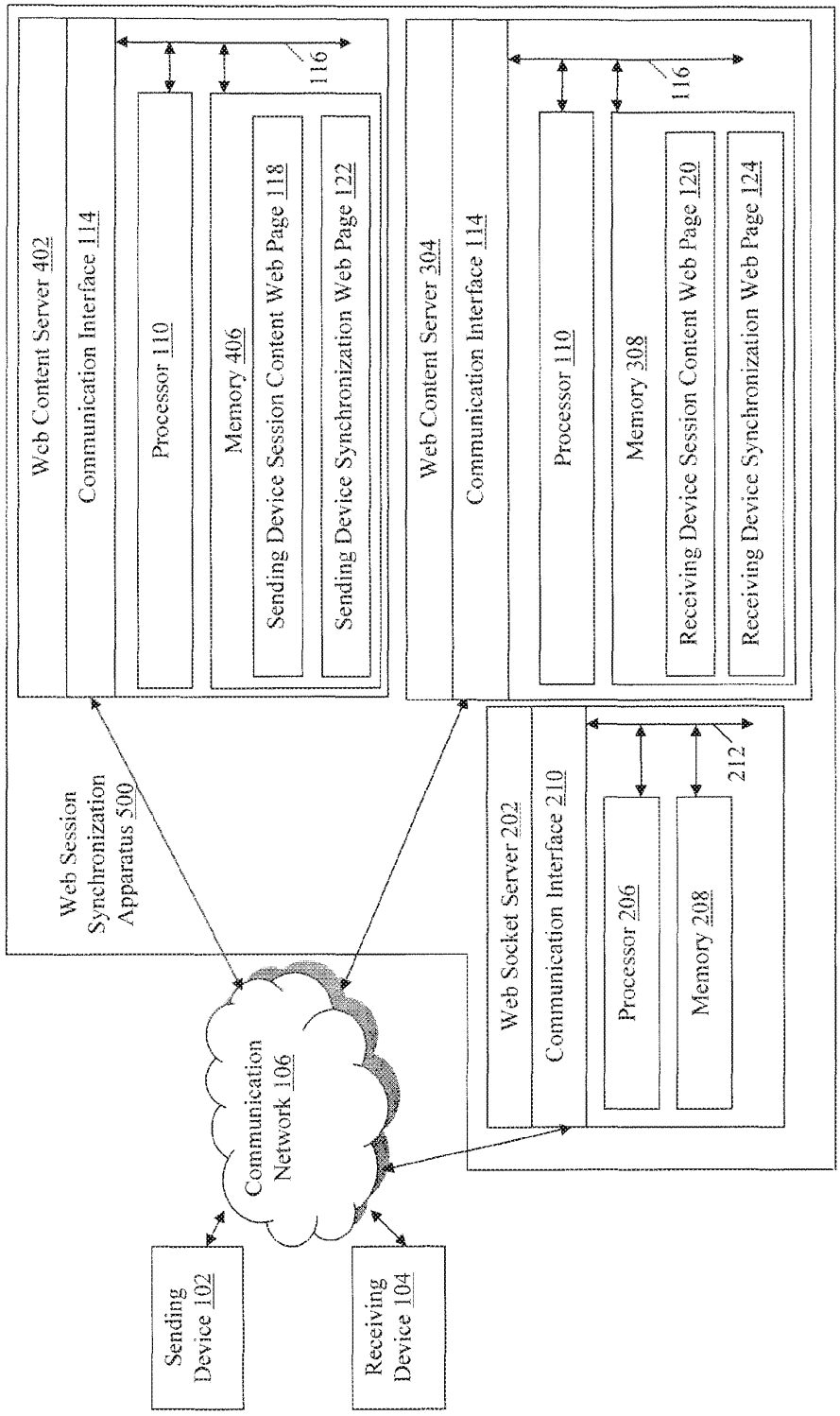

Web Content Server 402
Communication Interface 114
Processor 110
Memory 406
Sending Device Session Content Web Page 118
Sending Device Synchronization Web Page 122
116
Web Content Server 304
Communication Interface 114
Processor 110
Memory 308
Receiving Device Session Content Web Page 120
Receiving Device Synchronization Web Page 124
116
Web Session Synchronization Apparatus 500
Web Socket Server 202
Communication Interface 210
Processor 206
Memory 208
212
Communication Network 106
Sending Device 102
Receiving Device 104

FIG. 5

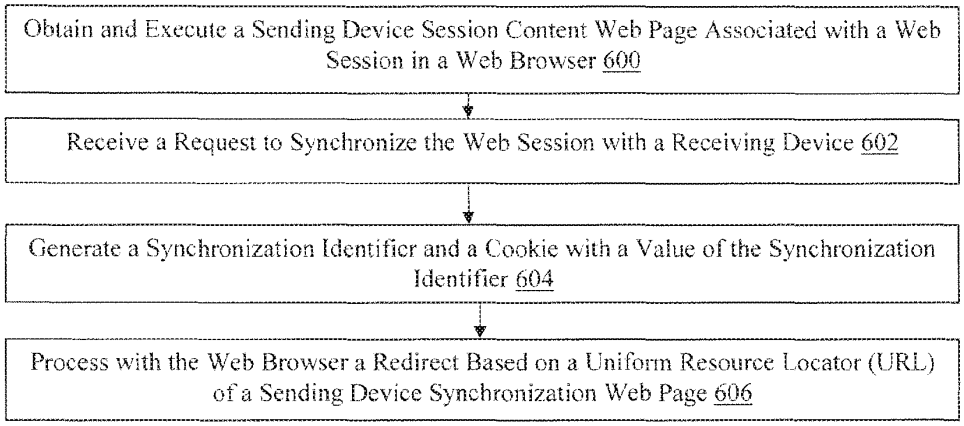

Obtain and Execute a Sending Device Session Content Web Page Associated with a Web Session in a Web Browser 600
Receive a Request to Synchronize the Web Session with a Receiving Device 602
Generate a Synchronization Identifier and a Cookie with a Value of the Synchronization Identifier 604
Process with the Web Browser a Redirect Based on a Uniform Resource Locator (URL) of a Sending Device Synchronization Web Page 606

FIG. 6

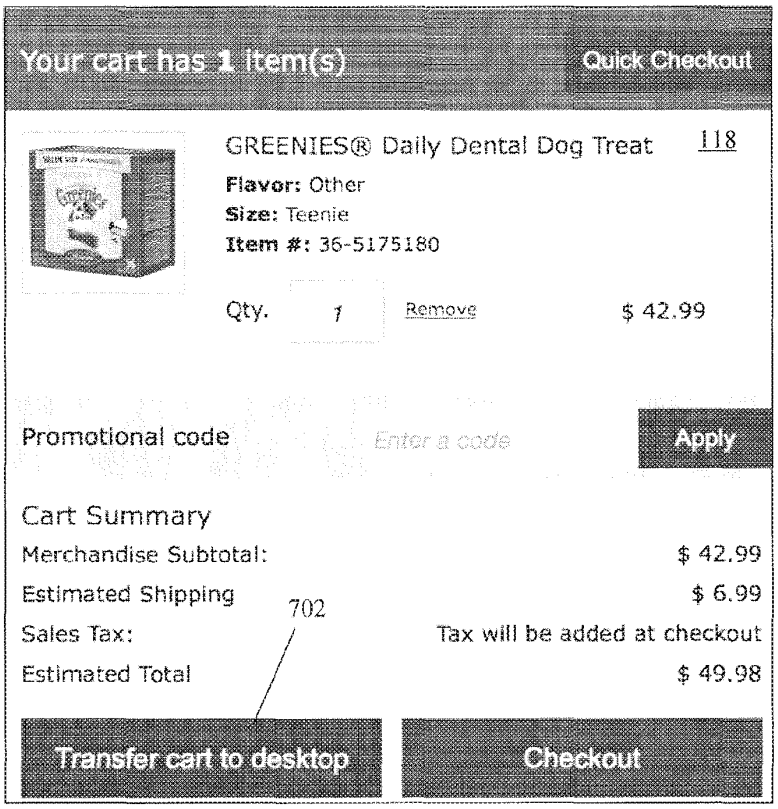

Your cart has 1 item(s)
Quick Checkout
GREENIES® Daily Dental Dog Treat
118
Flavor: Other
Size: Teenie
Item #: 36-5175180
Qty. 1
Remove
$ 42.99
Promotional code
Enter a code
Apply
Cart Summary
Merchandise Subtotal: $ 42.99
Estimated Shipping $ 6.99
Sales Tax: Tax will be added at checkout
Estimated Total $ 49.98
702
Transfer cart to desktop
Checkout

FIG. 7

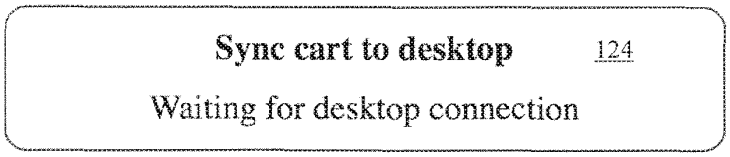


FIG. 8

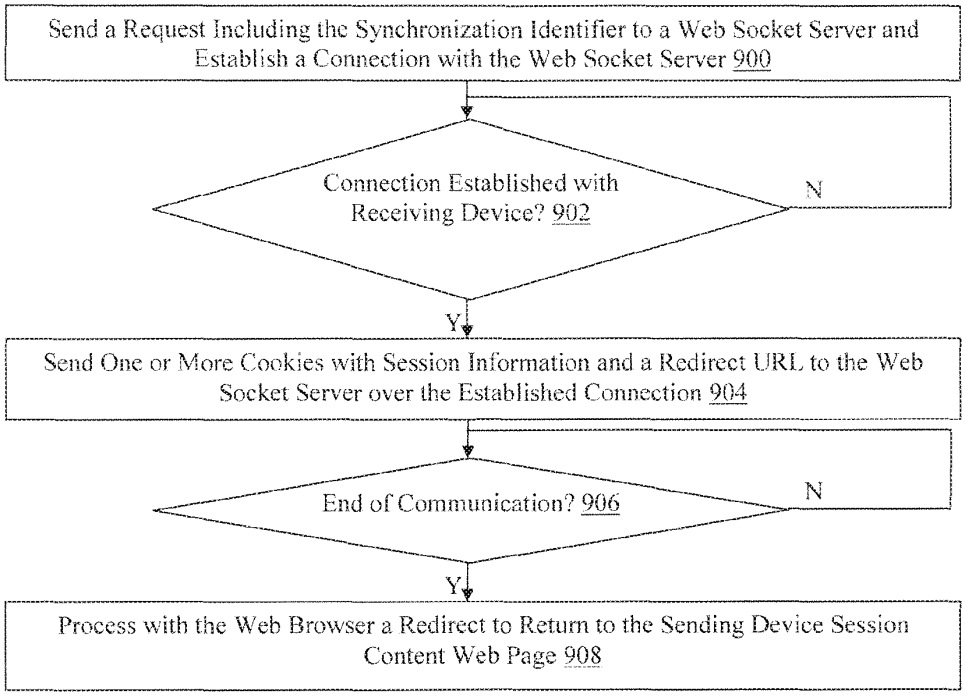


FIG. 9

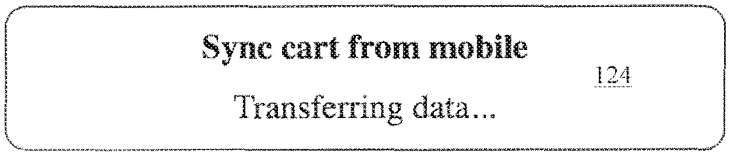


FIG. 10

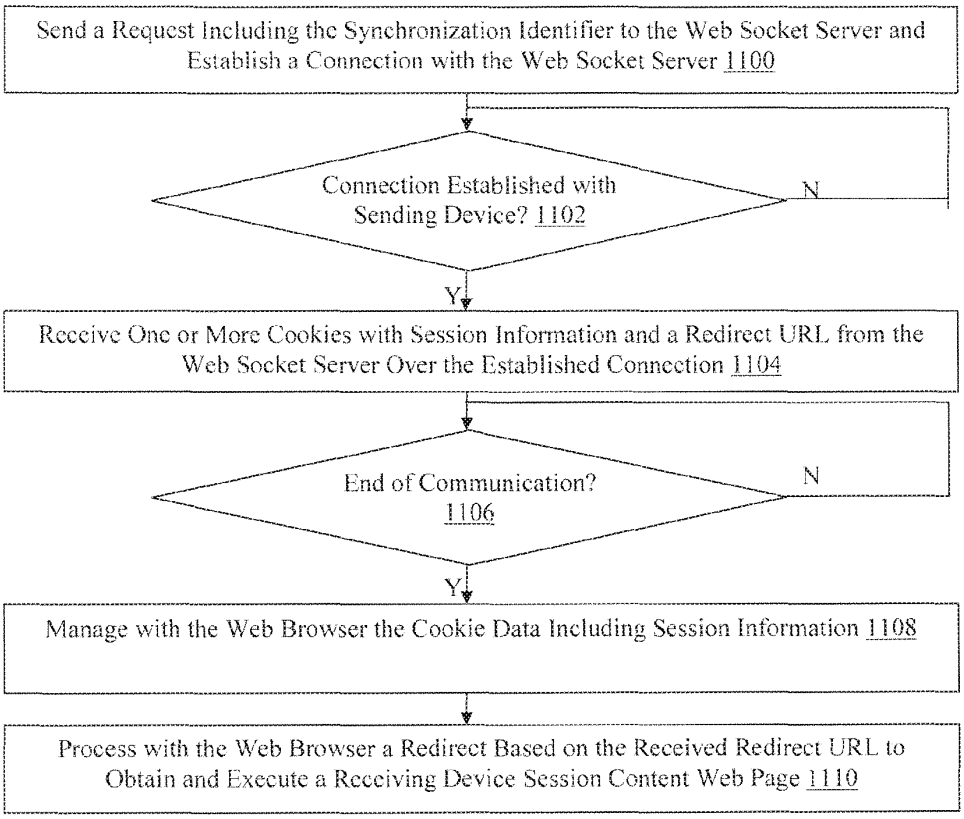

Send a Request Including the Synchronization Identifier to the Web Socket Server and Establish a Connection with the Web Socket Server 1100
Connection Established with Sending Device? 1102
N
Y
Receive One or More Cookies with Session Information and a Redirect URL from the Web Socket Server Over the Established Connection 1104
End of Communication? 1106
N
Y
Manage with the Web Browser the Cookie Data Including Session Information 1108
Process with the Web Browser a Redirect Based on the Received Redirect URL to Obtain and Execute a Receiving Device Session Content Web Page 1110

FIG. 11

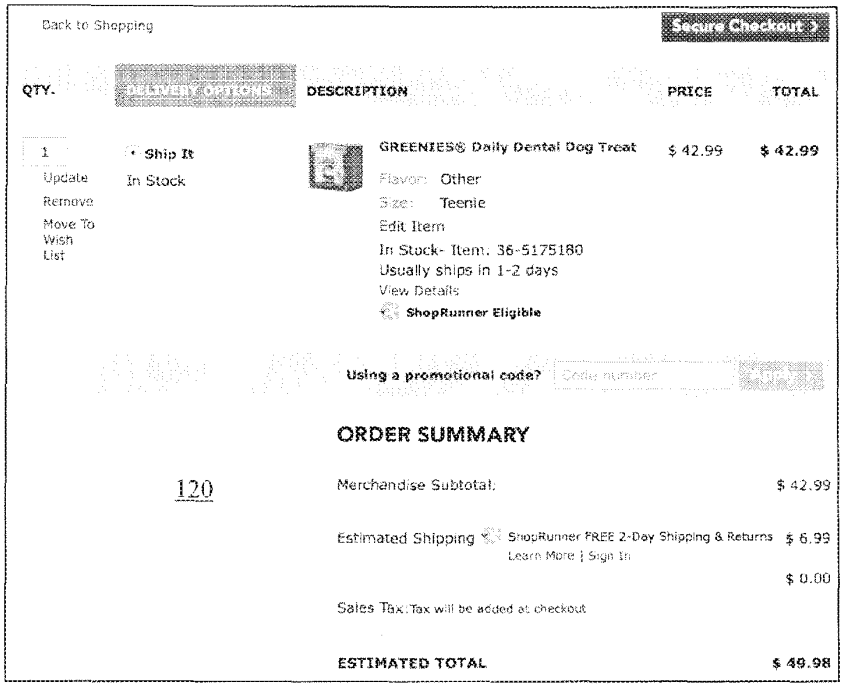

Back to Shopping
Secure Checkout >
QTY.
DELIVERY OPTIONS
DESCRIPTION
PRICE
TOTAL
1
Update
Remove
Move To Wish List
Ship It
In Stock
GREENIES® Daily Dental Dog Treat
$ 42.99
$ 42.99
Flavor: Other
Size: Teenie
Edit Item
In Stock- Item: 36-5175180
Usually ships in 1-2 days
View Details
ShopRunner Eligible
Using a promotional code?
Code number
Apply
ORDER SUMMARY
120
Merchandise Subtotal:
$ 42.99
Estimated Shipping
ShopRunner FREE 2-Day Shipping & Returns
Learn More | Sign In
$ 6.99
$ 0.00
Sales Tax: Tax will be added at checkout
ESTIMATED TOTAL
$ 49.98

FIG. 12

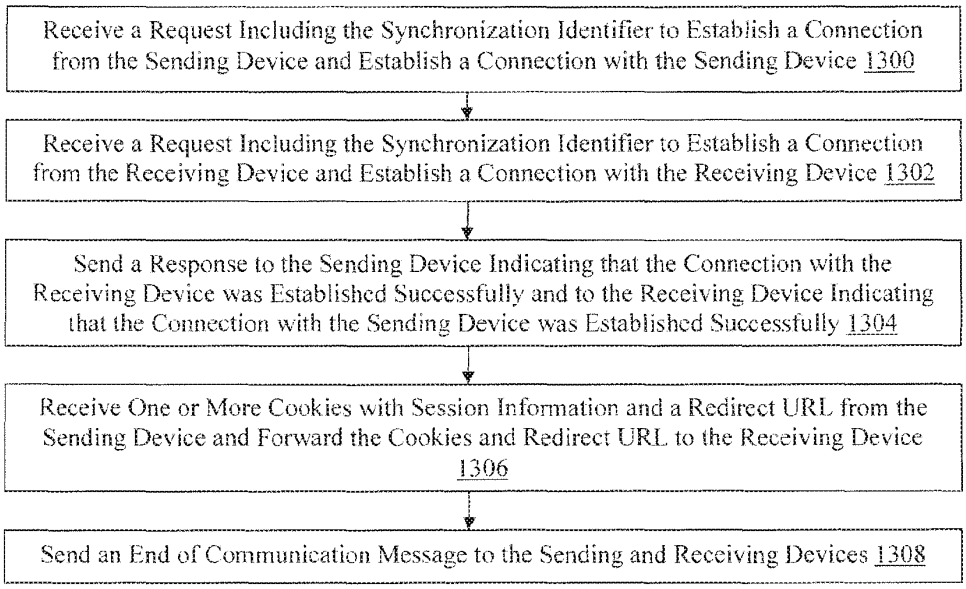

Receive a Request Including the Synchronization Identifier to Establish a Connection from the Sending Device and Establish a Connection with the Sending Device 1300
Receive a Request Including the Synchronization Identifier to Establish a Connection from the Receiving Device and Establish a Connection with the Receiving Device 1302
Send a Response to the Sending Device Indicating that the Connection with the Receiving Device was Established Successfully and to the Receiving Device Indicating that the Connection with the Sending Device was Established Successfully 1304
Receive One or More Cookies with Session Information and a Redirect URL from the Sending Device and Forward the Cookies and Redirect URL to the Receiving Device 1306
Send an End of Communication Message to the Sending and Receiving Devices 1308

FIG. 13

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 15/36717

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - G06F 15/16 (2015.01)
CPC - H04L 29/0854; H04L 29/08072; H04L 29/06
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
CPC: H04L 29/0854; H04L 29/08072; H04L 29/06; IPC(8): G06F 15/16 (2015.01)

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
USPC: 709/248; 709/227; 709/21 7; CPC: H04L 29/0854; H04L 29/08072; H 04L 29/06; H04L 29/0809, G06F 1/14; IPC(8): G06F 15/16 (2015.01) (keyword limited, terms below)

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
PatBase, Google Patents, IEEE; Search Terms: synchronizing; web session, web socket connection; token, cookie; uniform resource locator, URL, uniform; resource identifier, URI; notifying, forwarding

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2014/0164447 A1 (Tarafdar et al.) 12 June 2014 (12.04.2014), entire document especially paras [0042], [0043], [0044], [0045], [0046], [0051], [0057], [0058], [0060], [0092]	1 - 18
A	US 2009/0094688 A1 (Roy) 09 April 2009 (09.04.2009), entire document	1 - 18
A	US 2010/0281 107 A1 (Fallows et al.) 04 November 2010 (04.1 1.2010), entire document	1 - 18
A	US 2003/01 31045 A1 (McGee et al.) 10 July 2003 (10.07.2003), entire document	1 - 18
A	US 2009/0106349 A1 (Harris) 23 April 2009 (23.04.2009), entire document	1 - 18

☒ Further documents are listed in the continuation of Box C. ☐

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

Date of the actual completion of the international search	Date of mailing of the international search report
24 August 2015 (24.08.2015)	21 SEP 2015

Name and mailing address of the ISA/US	Authorized officer:
Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 2231 3-1450 Facsimile No. 571-273-8300	Lee W. Young PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

Form PCT/ISA/2 10 (second sheet) (January 20 15)

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
14 April 2016 (14.04.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/057092 A1

(51) International Patent Classification:
G06F 17/30 (2006.01)

(21) International Application Number:
PCT/US2015/036956

(22) International Filing Date:
22 June 2015 (22.06.2015)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
14/509,235 8 October 2014 (08.10.2014) US

(71) Applicant: **USABLENET INC.** [US/US]; 142 W. 57th Street, 7th Floor, New York, NY 10019 (US).

(72) Inventor: **SCODA, Enrico;** Via Cividina 416/3, Martignacco UD 33035 (IT).

(74) Agents: **GALLO, Nicholas, J.** et al; LeClairRyan, A Professional Corporation, 70 Linden Oaks, Suite 210, Rochester, NY 14625 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available):* AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available):* ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

(54) Title: METHODS FOR FACILITATING REFERENCES IN A CHAT CONTEXT AND DEVICES THEREOF

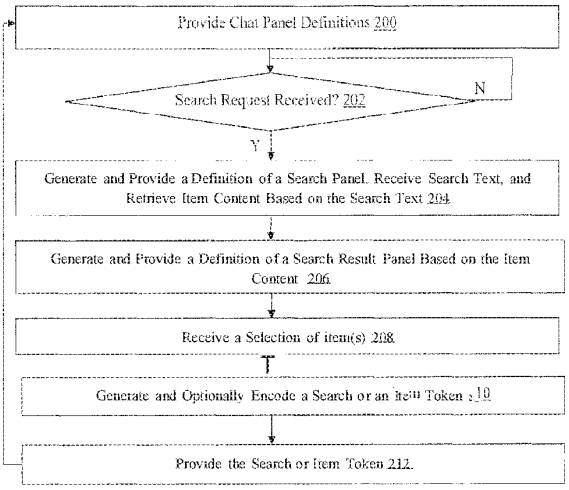


FIG. 2

(57) Abstract: A method, non-transitory computer readable medium, and chat management server apparatus that receives a search request via a search panel provided in response to a user interaction with a chat panel. A token including a special character is generated based on search text in the search request or a unique identifier for one of a plurality of items identified based on a search performed using the search text. The token is provided to a source of the search request for inclusion in the chat panel as a hyperlink. A preview panel request including the token is received in response to a user interaction with the hyperlink. Content for the items or for the one item is retrieved based on the special character included in the token. The content is provided to a source of the preview panel request.

WO 2016/057092 A1

METHODS FOR FACILITATING REFERENCES IN A CHAT CONTEXT AND DEVICES THEREOF

FIELD

[0001] This technology generally relates to web-based chat contexts and, more particularly, to methods, non-transitory computer readable media, and apparatuses that facilitate the inclusion of references in chat panels.

BACKGROUND

[0002] Increasingly, web developers are providing chat functionality in websites via chat contexts that include chat panels that are displayed to a user as embedded within a web page or via a pop-up window, for example. A chat panel can allow interaction and communication between the user and a representative of the website host. Accordingly, such functionality is particularly useful for websites providing user support, although many other types of websites also implement chat contexts. In chat contexts, the speed of an exchange is often critical to an effective experience for users and, accordingly, it is preferable that messages are simple and short.

[0003] As one example, in a commercial website context, sales representatives may use chat panels to communicate with potential customers to answer questions regarding products or services in order to facilitate and increase sales. In this example, a sales representative may want to refer a prospective customer to content hosted on other portions of the website, such as product catalog content relating to products that might satisfy desired criteria communicated by the prospective customer.

[0004] In order to refer the prospective customer to the content, the sales representative may copy and paste Uniform Resource Locators (URLs) as hyperlinks. However, URLs are often very long and inconvenient for use in a chat context. While available services can process a URL and replace the URL with a relatively short link, the process is cumbersome and would still result in numerous hyperlinks and a relatively long message when the content is located at a number of URLs (e.g., corresponding to a number of different products). Additionally, upon selecting any of the hyperlinks, the prospective customer may be taken to a different web page in a new tab or window, which is inconvenient and does not allow the customer to preview products prior to navigating to a different web page associated with one of the products. Accordingly, there is currently no way for the representative of a website host to identify items

and provide preview content associated with the items to the prospective customer in an efficient and effective manner in a chat context.

SUMMARY

[0005] A method for facilitating references in a chat context includes receiving by a chat management server apparatus a search request via a search panel provided in response to a user interaction with a chat panel. A token including a special character is generated by the chat management server apparatus based on search text in the search request or a unique identifier for one of a plurality of items identified based on a search performed using the search text. The token is provided by the chat management server apparatus to a source of the search request for inclusion in the chat panel as a hyperlink. A preview panel request including the token is received by the chat management server apparatus in response to a user interaction with the hyperlink. Content for the items or for the one item is retrieved by the chat management server apparatus based on the special character included in the token. The content is provided by the chat management server apparatus to a source of the preview panel request.

[0006] A non-transitory computer readable medium having stored thereon instructions for facilitating references in a chat context comprising executable code which when executed by a processor, causes the processor to perform steps including receiving a search request via a search panel provided in response to a user interaction with a chat panel. A token including a special character is generated based on search text in the search request or a unique identifier for one of a plurality of items identified based on a search performed using the search text. The token is provided to a source of the search request for inclusion in the chat panel as a hyperlink. A preview panel request including the token is received in response to a user interaction with the hyperlink. Content for the items or for the one item is retrieved based on the special character included in the token. The content is provided to a source of the preview panel request.

[0007] A chat management server apparatus including a processor and a memory coupled to the processor which is configured to be capable of executing programmed instructions comprising and stored in the memory to receive a search request via a search panel provided in response to a user interaction with a chat panel. A token including a special character is generated based on search text in the search request or a unique identifier for one of a plurality of items identified based on a search performed using the search text. The token is provided to a source of the search request for inclusion in the chat panel as a hyperlink. A preview panel request including the token is received in response to a user interaction with the hyperlink.

Content for the items or for the one item is retrieved based on the special character included in the token. The content is provided to a source of the preview panel request.

[0008] This technology provides a number of advantages including methods, non-transitory computer readable media, and apparatuses that more effectively facilitate references in a chat context. With this technology, website host representatives can identify content responsive to a search request from a user and communicate a reference to the content using a relatively short token. The token includes a special character and is introduced to a chat panel as a hyperlink. Upon selection of the hyperlink, the content is retrieved based on the token and included special character, and a preview panel is display to the user that includes the content.

[0009] Accordingly, using the tokens, the size of the communicated reference(s) can be reduced, particularly when multiple URLs would otherwise have been required to communicate references to content associated with multiple items responsive to a search request. Additionally, the user does not have to navigate away from the current web page to see the content. Moreover, the tokens can be reused, advantageously allowing the host representative to respond relatively quickly to certain search requests matching previously searched criteria.

BRIEF DESCRIPTION OF THE DRAWINGS

[0010] FIG. 1 is a block diagram of a network environment which incorporates an exemplary chat management server apparatus;

[0011] FIG. 2 is a flowchart of an exemplary method of generating an item preview panel based on a token;

[0012] FIG. 3 is an exemplary product web page with an exemplary chat panel link;

[0013] FIG. 4 is an exemplary host chat panel with a search request button;

[0014] FIG. 5 is an exemplary search panel for receiving search text;

[0015] FIG. 6 is an exemplary search result panel displaying content for a plurality of selectable items identified based on search text;

[0016] FIG. 7 is the exemplary chat panel of FIG. 4 with an item set link corresponding to a search token;

[0017] FIG. 8 is a flowchart of an exemplary method of generating a token for inclusion in a customer chat panel as a reference to item content;

[0018] FIG. 9 is an exemplary customer chat panel with the item set link and an exemplary multi-item preview panel with a navigational structure; and

[0019] FIG. 10 is the exemplary chat panel of FIG. 9 with an item link and an exemplary single item preview panel.

DETAILED DESCRIPTION

[0020] An exemplary network environment 10 is illustrated in FIG. 1 as including an exemplary chat management server apparatus 12. In this example, the chat management server apparatus 12 is coupled to a host representative device 14 by a local area network (LAN) 16 and a client device 18 by the LAN 16 and a wide area network (WAN) 20, although other types and numbers of devices, components, and elements in other topologies could be used. This technology provides a number of advantages including methods, non-transitory computer readable media, and apparatuses that more efficiently and effectively facilitate identifying and providing references to content matching user search criteria in a chat context.

[0021] In this example, the chat management server apparatus 12 includes a processor 22, a memory 24, and an interface device 26, which are coupled together by a bus 28 or other communication link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor 22 of the chat management server apparatus 12 may execute one or more stored programmed instructions for one or more aspects of this technology as described and illustrated by way of the embodiments herein, although the processor 22 could execute other numbers and types of programmed instructions.

[0022] The memory 24 of the chat management server apparatus 12 stores these programmed instructions for one or more aspects of this technology, as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. The memory 24 of the chat management server apparatus 12 may include one or more tangible storage media and/or devices, such as RAM, ROM, flash memory, hard disk drive(s), solid state memory, or any other memory storage types or devices, including combinations thereof, which are known to those of ordinary skill in the art.

[0023] In this example, the memory 24 of the chat management server apparatus 12 includes an item content catalog 30, a search web service 32, a preview web service 34, and an optional encoded token database 36, although the memory 24 can include other types and numbers of systems, devices, and elements in other configurations. Additionally, while the item content catalog 30, search web service 32, preview web service 34, and encoded token database 36 are illustrated in this example as being stored in the memory 24 of the chat management server apparatus 12, one or more of the item content catalog 30, search web service 32, preview web service 34, or encoded token database 36 could be stored elsewhere, including on another network device not shown in the network environment 10.

[0024] The item content catalog 30 in this example includes content for items, which can represent products or services, for example. The content can include a unique identifier for the item and information regarding the item including a description of the item, an item price, item options (e.g., colors), and/or any pictures or graphics associated with the item, for example, although any other type of content can also be stored in the item content catalog.

[0025] The search web service 32 in this example is configured to receive a request including search criteria and to identify matching item(s) in the item content catalog. The criteria can include search text, for example, when associated with a request to generate a search panel or a search token, for example, when associated with a request to generate a preview panel, as described and illustrated in more detail later. In response to the request, the search web service 32 retrieves at least a portion of the content (e.g., as used to present a preview of the item to a user) for each of the identified item(s) from the item content catalog 30 and generates and returns a HyperText Markup Language (HTML) fragment including the content, as described and illustrated in more detail later.

[0026] The preview web service 34 in this example is configured to receive a request to generate a preview panel that includes an item token. In response to the request, the preview web service 34 retrieves at least a portion of the content associated with an item identified based on the item token from the item content catalog 30 and generates and returns an HTML fragment including the content, as described and illustrated in more detail later.

[0027] The optional encoded token database 36 in this example stores encoded token values as associated with actual values. The actual values can be search text in the case of an encoded search token value or a unique item identifier in the case of an encoded item token value. As described and illustrated in more detail later, tokens can advantageously be encoded

with this technology to facilitate the inclusion of references in a chat panel that are even shorter than search text or item identifiers, for example.

[0028] The interface device 26 in the chat management server apparatus 12 is used to operatively couple and communicate between the chat management server apparatus 12 and the client device 18 via LAN 16 and WAN 20 and the host representative device 14 via the LAN 16, although other types and numbers of communication networks or systems with other types and numbers of connections and configurations to other devices and elements can also be used. The LAN 16 and WAN 20 can use TCP/IP over Ethernet and industry-standard protocols, including NFS, CIFS, SOAP, XML, LDAP, and SNMP, for example, although other types and numbers of communication networks can also be used.

[0029] The client device 18 in this example enables a user to request, receive and interact with services and content hosted by the chat management server apparatus 12 via the LAN 16 and WAN 20, although the client device 18 could access content and utilize other types and numbers of content or applications from other sources and could provide a wide variety of other functions for a user. By way of example only, the client device 16 can be a mobile computing device, smart phone, personal digital assistant, or computer, for example.

[0030] The client device 18 includes a processor 38, a memory 40, an interface device 42, an input device 43, and a display device 44 which are coupled together by a bus 45 or other communication link, although the client device 18 can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 38 in the client device 18 executes a program of stored instructions for one or more aspects of the present invention as described and illustrated herein, although the processor 38 could execute other numbers and types of programmed instructions.

[0031] The memory 40 in the client device 18 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a RAM, ROM, hard disk drive(s), solid state storage device(s), and/or other storage device which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor 38 can be used for the memory 40 in the client device 18. In this example, the client device 18 is configured to access web services and web content through a web browser 46 stored in the memory 40. The web browser 46 in this example is configured to process programmed instructions (e.g., JavaScript

code) to render chat panels and preview panels, as well as provide other functionality, as described and illustrated in more detail later.

[0032] The interface device 42 in the client device 18 is used to operatively couple and communicate between the client device 18 and the chat management server apparatus 12 via the LAN 16 and the WAN 20, although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[0033] The input device 43 in the client device 18 can be used to input selections, such as a request for a chat or preview panel, as well as messages to be exchanged with the user of the host representative device 14, although the input device 43 could be used to input other types of data and interact with other elements. The input device 43 can include keypads, touch screens, and/or vocal input processing systems, although other types and numbers of input devices can also be used.

[0034] The display device 44 in the client device 18 can be used to show data and information to a user, such as the requested chat or preview panel, although the display device 44 could be used to display other types of data and interact with other elements. The display device 44 can be television screen, a mobile phone screen display, a laptop screen, a tablet screen, or a monitor for example, although other types and numbers of displays could be used depending on the particular type of client device 18.

[0035] The host representative device 14 includes a processor 48, a memory 50, an interface device 52, an input device 53, and a display device 54, which are coupled together by a bus 55 or other communication link, although the host representative device 14 can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 48 in the host representative device 14 executes a program of stored instructions for one or more aspects of the present invention as described and illustrated herein, although the processor 48 could execute other numbers and types of programmed instructions.

[0036] The memory 50 in the host representative device 14 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a RAM, ROM, hard disk drive(s), solid state storage device(s), and/or other storage device which is read from and/or

written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor 38 can be used for the memory 40 in the host representative device 14.

[0037] In this example, the host representative device 14 is operated by a representative of a host of the website associated with the item content, although the host representative device 14 could be operated by other users, in order to engage in a chat with a user of the client device 18. Accordingly, the memory 50 in this example includes a web browser 56 through which the user of the host representative device can access web services and web content. The web browser 56 in this example is configured to process programmed instructions (e.g., JavaScript code) to render chat panels, search panels, and search result panels, as well as provide other functionality, as described and illustrated in more detail later.

[0038] The interface device 52 in the host representative device 14 is used to operatively couple and communicate between the host representative device 14 and the chat management server apparatus 12 via the LAN 16, although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[0039] The input device 53 in the host representative device 18 can be used to input selections, such as a request for a search panel, as well as messages to be exchanged with the user of the client device 18, although the input device 53 could be used to input other types of data and interact with other elements. The input device 53 can include keypads, touch screens, and/or vocal input processing systems, although other types and numbers of input devices can also be used.

[0040] The display device 54 in the host representative device 18 can be used to show data and information to a user, such as the requested search panel, although the display device 54 could be used to display other types of data and interact with other elements. The display device 54 can be television screen, a mobile phone screen display, a laptop screen, a tablet screen, or a monitor for example, although other types and numbers of displays could be used depending on the particular type of host representative device 18.

[0041] Although embodiments of the chat management server apparatus 12, host representative device 14, and client device 18 are described and illustrated herein, each of these devices can be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the

embodiments are possible, as will be appreciated by those skilled in the relevant art(s). Furthermore, each of the systems of the embodiments may be conveniently implemented using one or more general purpose computer systems, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[0042] In addition, two or more computing systems or devices can be substituted for any one of the devices in any embodiment. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the embodiments. The embodiments may also be implemented on computer system(s) that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[0043] The examples may also be embodied as a non-transitory computer readable medium having instructions stored thereon for one or more aspects of the present technology as described and illustrated by way of the examples herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the examples, as described and illustrated herein.

[0044] An exemplary method for facilitating references in a chat context will now be described with reference to FIGS. 1-10. Referring specifically to FIG. 2, an exemplary method of generating a token for inclusion in a chat panel as a reference to item content will now be described. In this example, in step 200, the chat management server apparatus 12 provides chat panel definitions in response to a request received from the client device 18. Referring more specifically to FIG. 3, an exemplary product web page 300 with an exemplary chat panel link 302 is illustrated. In this example, a user of the client device 18 use the input/display device 43 to select the chat panel link 302 thereby initiating a chat session and requesting a customer chat panel from the chat management server apparatus 12, although the chat session can be initiated in other manners.

[0045] In response, the chat management server apparatus 12 sends a definition of a customer chat panel to the client device 18 and a host chat panel to the host representative device

14. The definitions can define the chat panels using HTML, for example, which, when interpreted by the web browsers 46 and 56 cause the client device 18 and host representative device 14 to display the customer and host chat panels, respectively, in a pop-up window or an overlay, for example, although any other type of display for the chat panels can also be used. The chat panels facilitate communication of messages between the users of the client device 18 and host representative device 14.

[0046] Referring more specifically to FIG. 4, an exemplary host chat panel 400 is illustrated. In this example, a definition of the host chat panel 400 is provided by the chat management server 12 to the host representative device 14 in response to receiving an initiation of the chat session in step 200 from the client device 18. The definition sent by the chat management server 12 to the client device 18 in step 200 can define a customer chat panel similar to the host chat panel 400 but without the search request button 402, for example, although the chat panels can be similar or different in other ways and other methods of distributing and generating the chat panels can also be used. An exemplary customer chat panel will be described and illustrated in more detail later with reference to FIGS. 9-10.

[0047] In this example, the user of the client device 18 is a prospective customer of the host of the product web page 300 that is looking for assistance completing an outfit prior to purchasing, although the web page 300 can be any other type of web page and the user of the client device 14 can be any other type of user. Accordingly, the customer user of the client device 18 initiates the chat session in order to communicate with a representative of the host of the website, that is currently using the host representative device 14, to obtain the requested assistance in an efficient manner.

[0048] Referring back to FIG. 2, in step 202, the chat management server apparatus 12 determines when a search request has been received from the host representative device 14. The search request can be for content that a user of the host representative device 14 would like to refer the customer to in order to attempt to assist the customer and facilitate a purchase, for example. If the chat management server apparatus 12 determines that a search request has not been received, then the No branch is taken back to step 202 and the chat management server apparatus 12 effectively waits for a search request to be received.

[0049] Referring back to FIG. 4, the search request button 402 of the host chat panel 400 can be used to initiate a search request, although any other type of interface for initiating a search request can also be used. In this example, the customer user of the client device 18 is looking for

white shoes to pair with a specified dress. Accordingly, the host representative using the input/display device 53 of the host representative device 14 begins to type a message in an input field 404, having the text "No problem, here is a set of options for you:" in this example. Next, the host representative using the input/display device 53 of the host representative device 14 selects the search request button 402 in order to initiate a search for items responsive to the customer's request.

[0050] Referring back to step 202 of FIG. 2, upon selection of the search request button 402, or if the chat management server apparatus 12 otherwise determines that a search request has been received, then the Yes branch is taken to step 204. In step 204, the chat management server apparatus 12 generates, and provides to the host representative device 14, a definition of a search panel, receives search text, and retrieves item content based on the search text. The definition can define a search panel using HTML, for example, which, when interpreted by the web browser 56 can cause the host representative device 14 to display a search panel in a pop-up window or an overlay, for example, although the search panel can be displayed in other manners. The search panel is configured to receive search text, and optionally other search criteria, from the host representative using the host representative device 14.

[0051] Referring more specifically to FIG. 5, an exemplary search panel 500 for receiving search text via a text input box 502 is illustrated, although in other examples the search panel 500 can include inputs for other search criteria. In this example, the search text "faith court shoes white" is input by the host representative using the input/display device 53 of the host representative device 14. Upon selection of a search button 504 by the host representative, the search criteria are sent to the chat management server apparatus 12. In this example, the search criteria, including the search text, is sent to the search web service 32 of the chat management server apparatus 12, which is configured to process the criteria as described and illustrated in more detail later. Other methods of receiving search criteria can also be used.

[0052] In response to receiving the search text, the search web service 32 of the chat management server apparatus 12 identifies and retrieves item content responsive to the request in step 204. The item content can be retrieved from the item content catalog 30, which in this example includes content associated with a plurality of items for sale by the website host, although any other type of content associated with any other type of item can also be used. The content can include an item description, an item depiction, an item price, or any other information associated with each of the items. Optionally, the content includes at least

information suitable to provide the customer with a preview of the item to allow the customer to decide whether to learn more about the item in order to make a purchasing decision.

[0053] In step 206, the search web service 32 of the chat management server apparatus 12 generates and provides to the host representative device 14 a definition of a search result panel based on the item content retrieved in step 204. In this example, the item content can include a picture, a short description, and a price of various white faith court shoes identified based on the search text. Accordingly, the definition includes one or more HTML fragments for the identified item(s) that includes the content and is configured to generate a search result panel including the content when rendered by the web browser 56 of the host representative device 14.

[0054] Referring more specifically to FIG. 6, an exemplary search result panel 600 displaying content for a plurality of selectable items identified based on the search text is illustrated. In this example, the search result panel 600 includes item content 602(1) and 601(2) for two items ("white heeled court shoes" and "patent heeled court shoes") satisfying the search criteria received in step 204. Any number of items can be identified and included in the search result panel 600 and the search results can be displayed by the web browser 56 of the host representative device 14 in other manners.

[0055] Referring back to FIG. 2, in step 208, the chat management server 12 receives a selection from the host representative using the host representative device 14 of one or more of the item(s) for which content 602(1) and 601(2) was identified and retrieved in step 204, and provided to the host representative device 14 in step 206. The selected item(s) are those item(s) responsive to the customer's request and for which the host representative would like to specifically refer the customer to in order to assist the customer in making a purchasing decision in this example.

[0056] Referring back to FIG. 6, in this example, the content 602(1) and 602(2) for each item is associated with a select button 604(1) and 604(2), respectively. Additionally, the search result panel 600 includes a select all button 606. Upon selection of one of the buttons 604(1), 604(2), or 606 by the host representative using the input/display device 53 of the host representative device 14, an indication of the selection is sent to the chat management server apparatus 12. Accordingly, the definition of the search result panel 600 sent to the host representative device 14 in step 206 of FIG. 2 is configured to facilitate the selection of items, although other methods of facilitating the selection of item(s) can also be used.

[0057] In step 210, the chat management server apparatus 12 generates, and optionally encodes, a search or an item token. In this example, the tokens are prefixed by a first special character (e.g., "@") or a second special character (e.g., "#") according to whether the token is a search token or an item token, respectively, although the first or second special character can be included in the tokens in any location. The token is a search token if all of the items for which content is displayed in the search panel 600 are selected by the host representative using the host representative device 14 (e.g., by selecting the select all button 606). Additionally, the token is an item token if fewer than all of the items or which content is displayed in the search panel 600 are selected (e.g., using one or more of the select buttons 604(1) or 604(2)).

[0058] In this example, if the host representative using the host representative device 14 selects the select all button 606, an exemplary token including the first special character and the search text could be "@faith_court_shoes_white". By including the search text, the host representative can advantageously reuse this token in subsequent chat sessions as it will be relatively easy to remember. For example, the host representative can reuse the token with other prospective customers the host representative would like to refer to the same content, as described and illustrated in more detail later.

[0059] In another example, as described and illustrated in more detail later with reference to FIG. 10, if the host representative using the host representative device 14 selected only one of the items for which content was displayed on the search panel 600, an exemplary token including the second special character and a unique item identifier could be '#361 1369". Accordingly, in this example, the "361 1369" portion of the token corresponds to a unique identifier for the one selected item, as stored as associated with the content for the item in the content catalog 30.

[0060] However, in yet another example, the chat management server apparatus 12 can encode the token in order to reduce the size instead of merely using the search text or the unique item identifier for the portion of the token not including the special character. Accordingly, the chat management server apparatus 12 can decide to encode the token based on whether the number of characters in the search text or unique item identifier exceeds a threshold, for example, although the decision of whether to encode a token can be based on any other criteria. Additionally, the chat management server apparatus 12 can be configured to encode all or none of the tokens as a default setting.

[0061] If the chat management server apparatus 12 determines in step 210 that the search token in this example should be encoded, an exemplary encoded search token could be

"@42ad42", although any other encoding can be used. The chat management server apparatus 12 can replace the "faith_court_shoes_white" search text in the token by encoding the search text to generate an output of "42ad42". Any type of encoding function or formula can be used. Optionally, the output of the encoding can be limited to a certain number of characters in order to optimize the benefit of using an encoded token in place of a token including search text or an item identifier. Additionally, if the chat management server apparatus 12 determines in step 210 that the token should be encoded, then the chat management server apparatus 12 in this example stores at least the encoded portion of the search token as associated with the search criteria (e.g., the search text) in the encoded token database 36 so that the encoded token can subsequently be decoded, as described and illustrated in more detail later with reference to step 804 of FIG. 8.

[0062] Referring back to FIG. 2, in step 212, the chat management server apparatus 12 provides the search or item token to the host representative device 14. Referring more specifically to FIG. 7, the exemplary host chat panel 400 is illustrated with an item set link 702 corresponding to the search token "@42ad42", which is an encoded search token in this example. Accordingly, upon receipt by the host representative device 14 of the search token, the host chat panel 400 inserts the search token text into the input field 404.

[0063] Upon the host representative selecting the send button 700, using the input/display device 53 of the host representative device 14, the input text including the search token is sent to the chat management server 14, which routes the text to the customer chat panel currently rendered on the client device 18 using an established connection. Upon display of the input text in the host chat panel 400, as well as the customer chat panel currently rendered on the client device 18, the search token becomes the item set link 702.

[0064] Accordingly, the definition of the chat panel 400 sent to the host representative device 14 is configured to insert a token returned in step 212 into the input field 404 and render the token as a hyperlink (the item set link 702 in this example) used as described and illustrated in more detail with reference to FIG. 8. Optionally, at least the customer chat panel is configured, based on its definition, to render any text sent from a host representative in a chat session that includes the first or second special character as an item set link or an item link, respectively.

[0065] Referring more specifically to FIG. 8, an exemplary method of generating an item preview panel based on a token will now be described. In step 800 in this example, the chat management server apparatus 12 receives a request from the client device 18 for a preview panel.

The request includes a token and is sent in response to a selection by the customer, using the input/display device 43 of the client device 18, of a hyperlink including the token that was rendered in the customer chat panel. The hyperlink can be rendered in the customer chat panel subsequent to the host representative submitting a message including the token. The token can be manually entered by the host representative or provided by the chat management server apparatus 12, as described and illustrated earlier with reference to step 212.

[0066] In step 802, the chat management server apparatus 12 determines when the token included in the request for the preview panel received in step 800 is encoded. In order to determine whether the token is decoded, the chat management server apparatus 12 can compare the token to entries of the encoded token database 36 to determine where there is a match in this example, although other methods of determining whether the token is encoded can also be used. If the chat management server apparatus 12 determines that the token is encoded, then the Yes branch is taken to step 804.

[0067] In step 804, the chat management server apparatus 12 decodes the token. In order to decode the token in this example, the chat management server apparatus 12 retrieves the actual value of the portion of the token not including the special character from the matching entry of the encoded token database 36. The actual value could have been stored in the encoded token database 36 as described and illustrated in more detail earlier with reference to step 210 of FIG. 2. Accordingly, in this example, the chat management server 12 can obtain the "faith_court_shoes_white" actual value by decoding the "42ad42" encoded token value. Other methods of encoding or decoding the tokens, including using a reversible encoding function that does not require a database look-up, can also be used. Subsequent to decoding the token, or if the chat management server apparatus 12 determines in step 802 that the token is not encoded and the No branch is taken, the chat management server apparatus 12 proceeds to step 806.

[0068] In step 806, the chat management server apparatus 12 retrieves item content based on the token. In order to retrieve the item content, in this example, the chat management server apparatus 12 first determines whether the token is a search or an item token based on whether the token includes the first or second special character. If the chat management server apparatus 12 determines that the token is a search token, then the portion of the token not including the first special character, or the decoded actual value in examples in which step 804 is performed, is processed by the search web service 34. Accordingly, the search web service 32 searches the item content catalog 30, as described and illustrated earlier with reference to step 206 of FIG. 2, using the token value as the search text in order to identify and retrieve responsive item content.

[0069] However, if the chat management server apparatus 12 determines that the token is an item token, then the portion of the token not including the first special character, or the decoded actual value in examples in which step 804 is performed, is processed by the preview web service 34. Accordingly, the preview web service 34 searches the item content catalog 30 using the token value to identify and retrieve content for an item having a unique item identifier matching the token value.

[0070] In step 808, the chat management server apparatus 12 provides a preview panel definition, including HTML fragment(s) including the content retrieved in step 806, to the client device 18 in response to the request for the preview panel received in step 800. The preview panel definition is configured to, when rendered, cause the web browser 46 of the client device 18 to generate a preview panel that includes the item content. The preview panel can be generated a pop-up window, an overlay, or any other type of display that does not require navigation by the web browser 46 away from the customer chat panel. Additionally, the preview panel can be a multi-item preview panel with a navigation structure or a single item preview panel based on whether the token, received with the request for the preview panel in step 800, is a search token or an item token, respectively.

[0071] Optionally, at least a portion of the content displayed by the preview panel is, or another portion of the preview panel includes, a link that is selectable by the customer using the client device 18 in order to allow the customer to navigate to a different web page associated with the item that provides additional content. Also optionally, at least a portion of the content displayed by the preview panel is, or another portion of the preview panel includes, a link that is selectable by the customer using the client device 18 in order to allow the customer to navigate to an item purchase web page or add the item to a shopping cart, for example. Other types of links and other content can also be provided in the preview panel.

[0072] Referring more specifically to FIG. 9, an exemplary customer chat panel 900 with the item set link 702 corresponding to the encoded search token "@42ad42" and an exemplary multi-item preview panel 902 are illustrated. In this example, the multi-item preview panel 902 is generated, based on the definition provided in step 808, subsequent to the customer selecting the item set link 702 using the input/display device of the client device 18. The multi-item preview panel 902 includes the content 602(1) for one of the items that the host representative selected to be referred to the customer in this example.

[0073] Additionally, the multi-item preview panel 902 includes a navigation structure, which in this example includes a next button 904, which facilitates navigation between content associated with a plurality of items, including at least the content 602(2) of another of the items that the host representative selected to be referred to the customer in this example. Other types of navigational structures can also be used.

[0074] Referring more specifically to FIG. 10, the exemplary customer chat panel 900 of FIG. 9 with an item link 1000 and an exemplary single item preview panel 1002 are illustrated. In this example, the item link 1000 corresponds to an item token "#361 1369" which includes a unique item identifier "361 1369" for a single item selected by the host representative in an iteration of steps 202-212 of FIG. 2 performed prior to the example iteration described and illustrated in detail earlier. The item preview panel 1002 includes content 1004 for a navy colored shoe item that the customer in this example is not interested in. Instead, the customer indicated to the host representative a preference for a white shoe and the host representative submitted the request received in step 202 in the example iteration described and illustrated earlier in order to identify items responsive to the customer's preference.

[0075] Accordingly, with this technology, representatives of website hosts can more easily and effectively refer website users to preview content for items in a chat context. The references can be sent using tokens which are generally, or can be encoded to be, shorter than URLs associated with web pages corresponding to the items. Additionally, host representatives can refer users to preview content for item(s) by reusing tokens thereby facilitating relatively quick responsiveness. Moreover, items identified by a host representative can advantageously be displayed by a user in a preview panel without requiring the user to navigate away from the chat panel or the current web page and thereby significantly improving the functioning of the user's client device.

[0076] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for facilitating references in a chat context, the method comprising:

receiving, by a chat management server apparatus, a search request via a search panel provided in response to a user interaction with a chat panel;

generating, by the chat management server apparatus, a token including a special character and generated based on search text in the search request or a unique identifier for one of a plurality of items identified based on a search performed using the search text;

providing, by the chat management server apparatus, the token to a source of the search request for inclusion in the chat panel as a hyperlink;

receiving, by the chat management server apparatus, a preview panel request in response to a user interaction with the hyperlink, the preview panel request including the token;

retrieving, by the chat management server apparatus, content for the items or for the one item based on the special character included in the token; and

providing, by the chat management server apparatus, the content to a source of the preview panel request.

2. The method as set forth in claim 1, further comprising, prior to generating the token:

retrieving, by the chat management server apparatus, the content for the items based on the search text;

providing, by the chat management server apparatus, the content for the items to the source of the search request for inclusion in a search result panel; and

receiving, by the chat management server apparatus, a selection of one or more of the items from the source of the search request.

3. The method as set forth in claim 2, wherein the selection is of all of the items and the generating further comprises generating a search token comprising a first special character and the request for the preview panel includes the search token.

4. The method as set forth in claim 2, wherein the selection is of one or more

of the items, the generating further comprises generating an item token for each of the one or more items, each item token comprises a second special character, and the request for the preview panel includes at least one of the item tokens.

5. The method as set forth in claim 3, wherein the search token includes at least one or more terms included in the search text.

6. The method as set forth in claim 1, wherein:

the generating further comprises encoding the search text or the unique identifier and storing the encoded search text or the encoded unique identifier in an encoded token database as associated with the corresponding search text or unique identifier, wherein the token is generated based on the encoded search text or the encoded unique identifier; and

the retrieving further comprises decoding the token comprising comparing the token to the encoded token database to retrieve the search text or unique identifier.

7. The method as set forth in claim 1, wherein the retrieving further comprises:

determining when the token included in the preview panel request is a search token based on a match of a specified character of the token with a first special character;

performing a search of an item content catalog using the search text, as determined based on the token included in the preview panel request , to retrieve the content for the items, when the token included in the preview panel request is determined to be the search token; and

retrieving content for the one item based on the unique identifier, as determined based on the token included in the preview panel request, when the token included in the preview panel request is not determined to be the search token.

8. A non-transitory computer readable medium having stored thereon instructions for facilitating references in a chat context comprising executable code which when executed by a processor, causes the processor to perform steps comprising:

receiving a search request via a search panel provided in response to a user interaction with a chat panel;

generating a token including a special character and generated based on search text in the search request or a unique identifier for one of a plurality of items identified

based on a search performed using the search text;

providing the token to a source of the search request for inclusion in the chat panel as a hyperlink;

receiving a preview panel request in response to a user interaction with the hyperlink, the preview panel request including the token;

retrieving content for the items or for the one item based on the special character included in the token; and

providing the content to a source of the preview panel request.

9. The non-transitory computer readable medium as set forth in claim 8, further having stored thereon instructions that when executed by the processor cause the processor to perform steps further comprising, prior to generating the search token or the item token:

retrieving the content for the items based on the search text;

providing the content for the items to the source of the search request for inclusion in a search result panel; and

receiving a selection of one or more of the items from the source of the search request.

10. The non-transitory computer readable medium as set forth in claim 9, wherein the selection is of all of the items and the generating further comprises generating a search token comprising a first special character and the request for the preview panel includes the search token.

11. The non-transitory computer readable medium as set forth in claim 9, wherein the selection is of one or more of the items, the generating further comprises generating an item token for each of the one or more items, each item token comprises a second special character, and the request for the preview panel includes at least one of the item tokens.

12. The non-transitory computer readable medium as set forth in claim 10, wherein the search token includes at least one or more terms included in the search text.

13. The non-transitory computer readable medium as set forth in claim 8, wherein:

the generating further comprises encoding the search text or the unique identifier and storing the encoded search text or the encoded unique identifier in an encoded token database as associated with the corresponding search text or unique identifier, wherein the token is generated based on the encoded search text or the encoded unique identifier; and

the retrieving further comprises decoding the token comprising comparing the token to the encoded token database to retrieve the search text or unique identifier.

14. The non-transitory computer readable medium as set forth in claim 8, wherein the retrieving further comprises:

determining when the token included in the preview panel request is a search token based on a match of a specified character of the token with a first special character;

performing a search of an item content catalog using the search text, as determined based on the token included in the preview panel request , to retrieve the content for the items, when the token included in the preview panel request is determined to be the search token; and

retrieving content for the one item based on the unique identifier, as determined based on the token included in the preview panel request, when the token included in the preview panel request is not determined to be the search token.

15. A chat management server apparatus, comprising a processor and a memory coupled to the processor which is configured to be capable of executing programmed instructions comprising and stored in the memory to:

receive a search request via a search panel provided in response to a user interaction with a chat panel;

generate a token including a special character and generated based on search text in the search request or a unique identifier for one of a plurality of items identified based on a search performed using the search text;

provide the token to a source of the search request for inclusion in the chat panel as a hyperlink;

receive a preview panel request in response to a user interaction with the hyperlink, the preview panel request including the token;

retrieve content for the items or for the one item based on the special character included in the token; and

provide the content to a source of the preview panel request.

16. The chat management server apparatus as set forth in claim 15, wherein the processor coupled to the memory is further configured to be capable of executing programmed instructions further comprising and stored in the memory to, prior to generating the search token or the item token:

retrieve the content for the items based on the search text;

provide the content for the items to the source of the search request for inclusion in a search result panel; and

receive a selection of one or more of the items from the source of the search request.

17. The chat management server apparatus as set forth in claim 16, wherein the selection is of all of the items, the processor coupled to the memory is further configured to be capable of executing at least one additional programmed instruction further comprising and stored in the memory to generate a search token comprising a first special character, and the request for the preview panel includes the search token.

18. The chat management server apparatus as set forth in claim 16, wherein the selection is of one or more of the items, the processor coupled to the memory is further configured to be capable of executing at least one additional programmed instruction further comprising and stored in the memory to generate an item token for each of the one or more items, each item token comprises a second special character, and the request for the preview panel includes at least one of the item tokens.

19. The chat management server apparatus as set forth in claim 17, wherein the search token includes at least one or more terms included in the search text.

20. The chat management server apparatus as set forth in claim 15, wherein the processor coupled to the memory is further configured to be capable of executing programmed instructions further comprising and stored in the memory to:

encode the search text or the unique identifier and storing the encoded search text or the encoded unique identifier in an encoded token database as associated with the corresponding search text or unique identifier, wherein the token is generated based on the encoded search text or the encoded unique identifier; and

decode the token comprising comparing the token to the encoded token database to retrieve the search text or unique identifier.

21. The chat management server apparatus as set forth in claim 15, wherein the processor coupled to the memory is further configured to be capable of executing programmed instructions further comprising and stored in the memory to:

determine when the token included in the preview panel request is a search token based on a match of a specified character of the token with a first special character;

perform a search of an item content catalog using the search text, as determined based on the token included in the preview panel request , to retrieve the content for the items, when the token included in the preview panel request is determined to be the search token; and

retrieve content for the one item based on the unique identifier, as determined based on the token included in the preview panel request, when the token included in the preview panel request is not determined to be the search token.

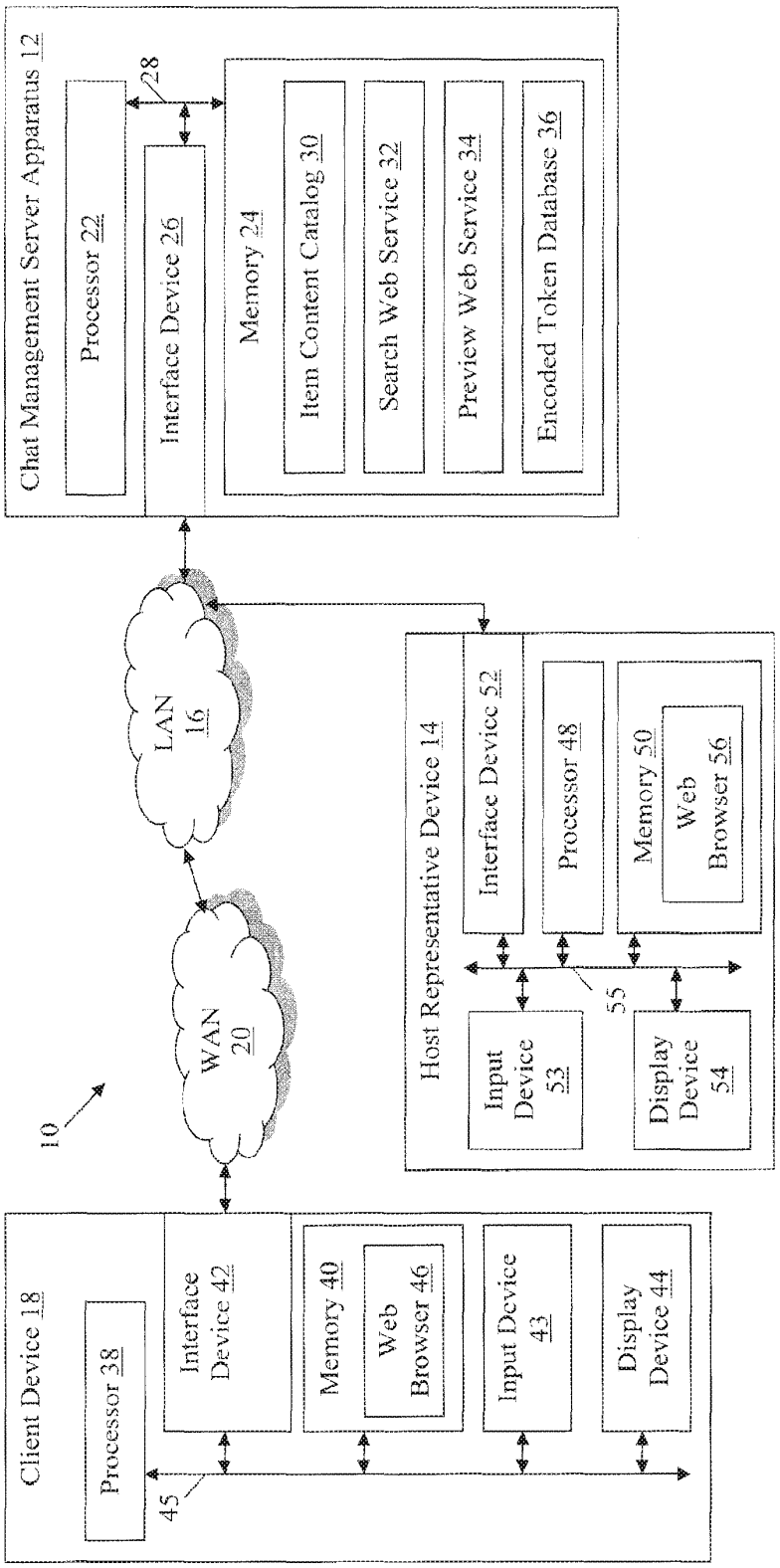


FIG. 1

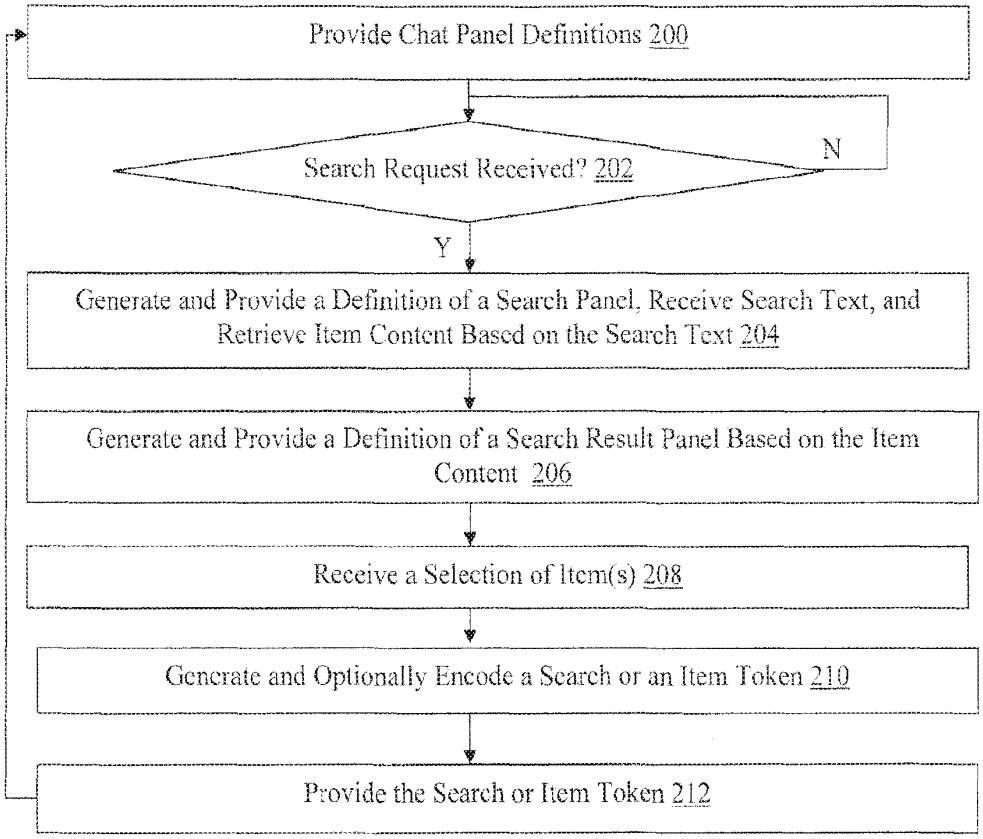


FIG. 2

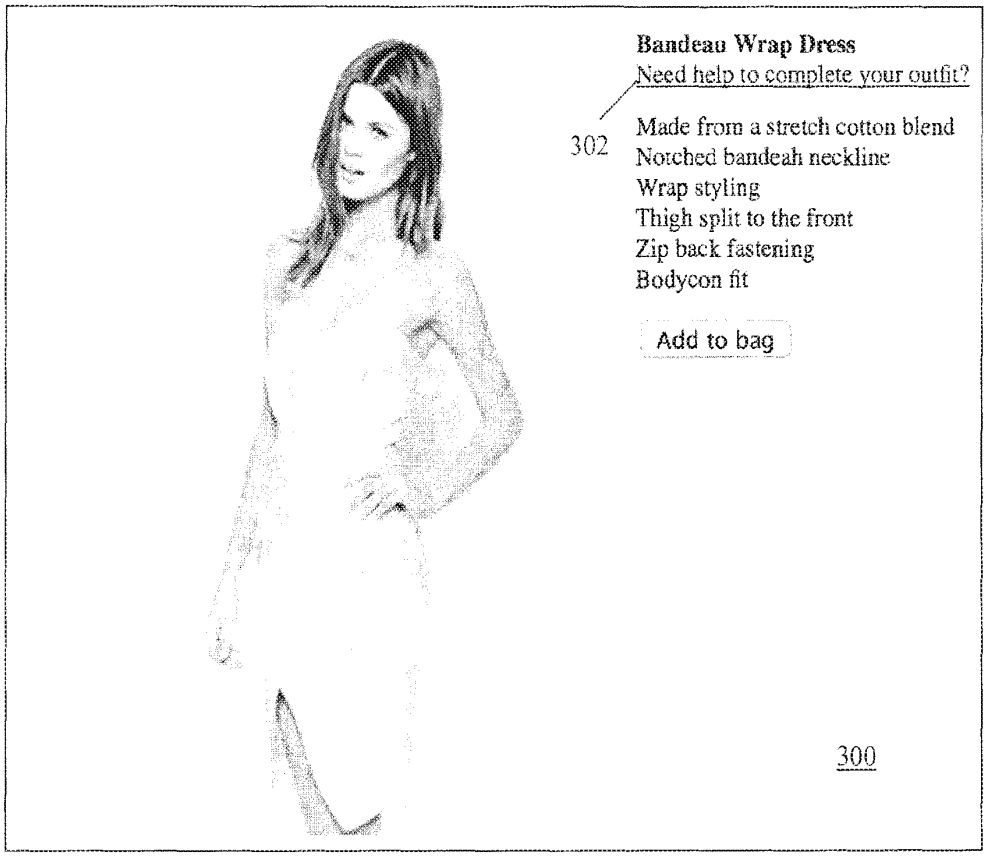


FIG. 3

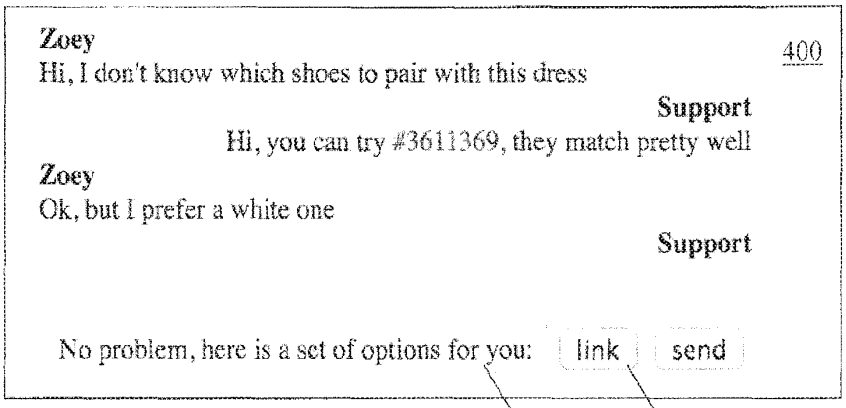


FIG. 4

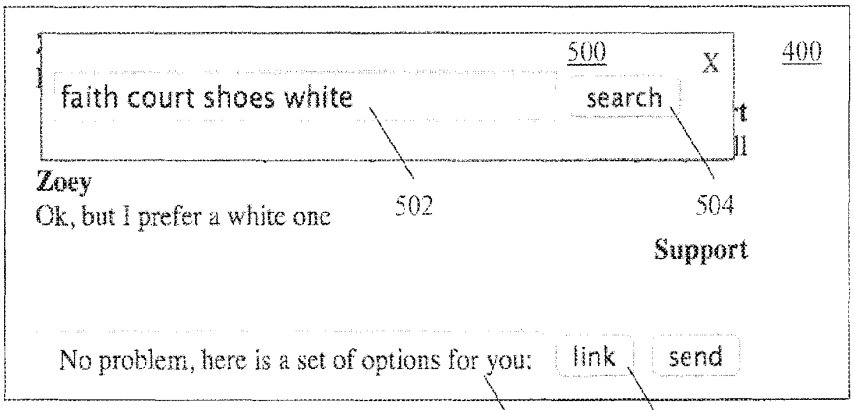


FIG. 5

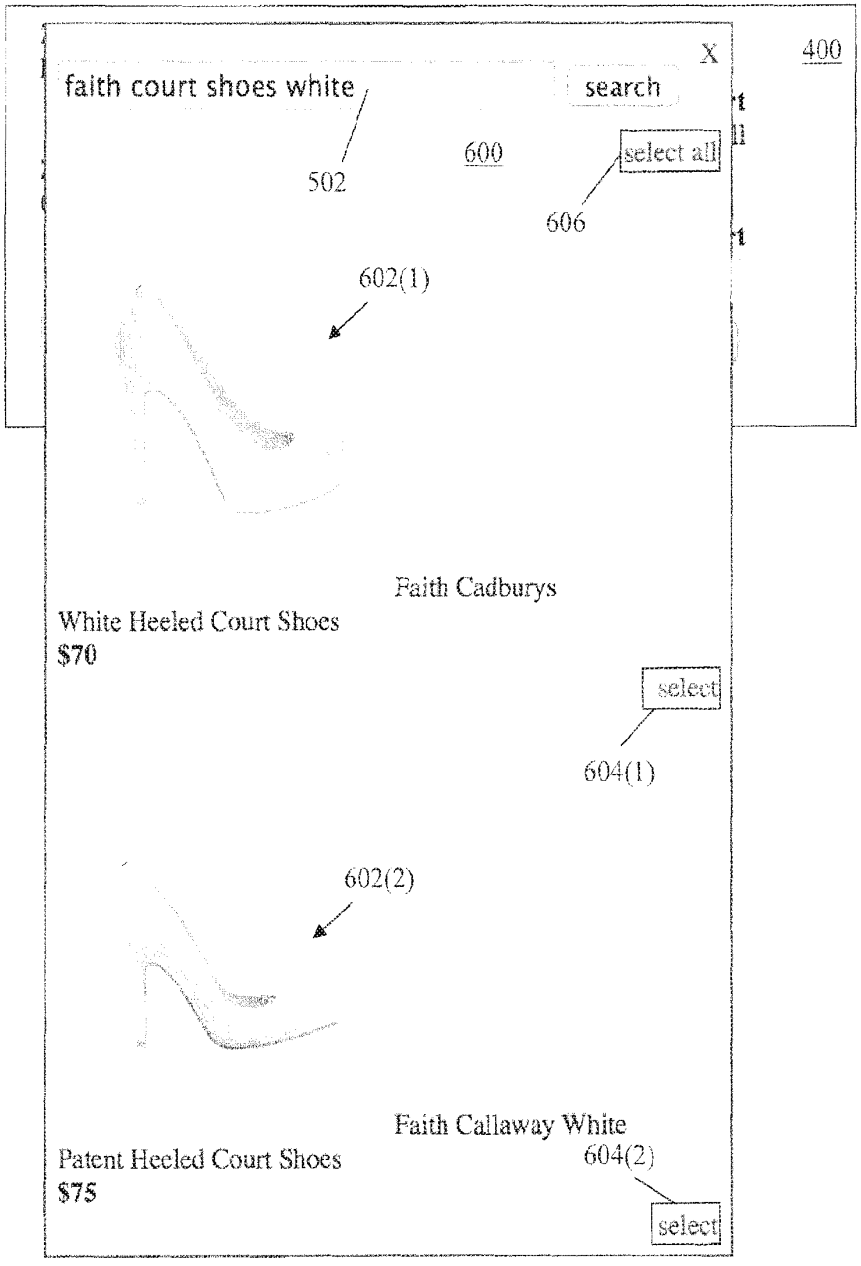


FIG. 6

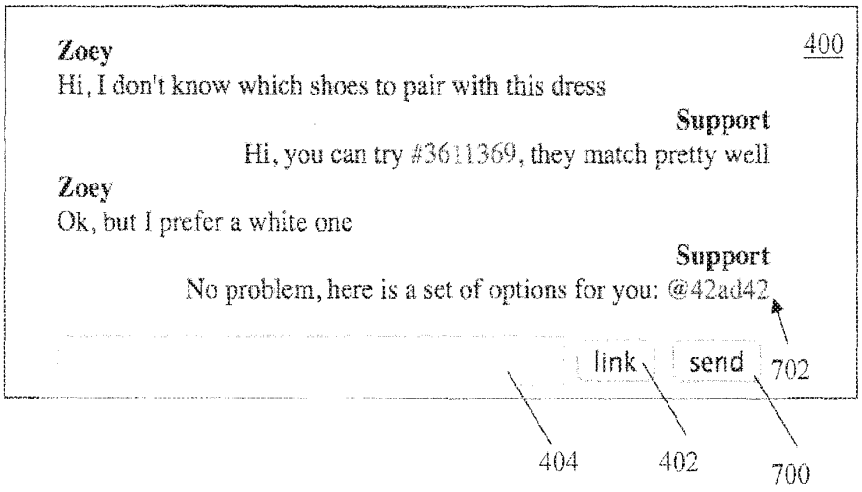


FIG. 7

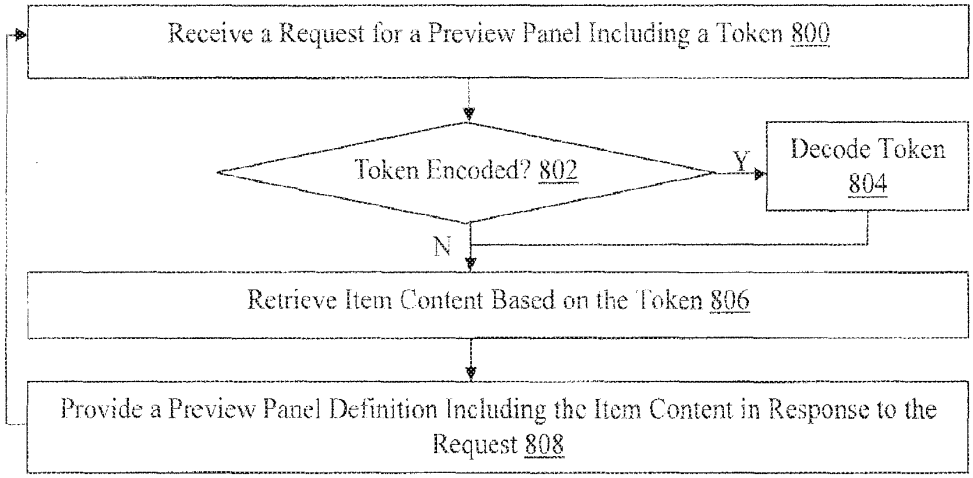


FIG. 8

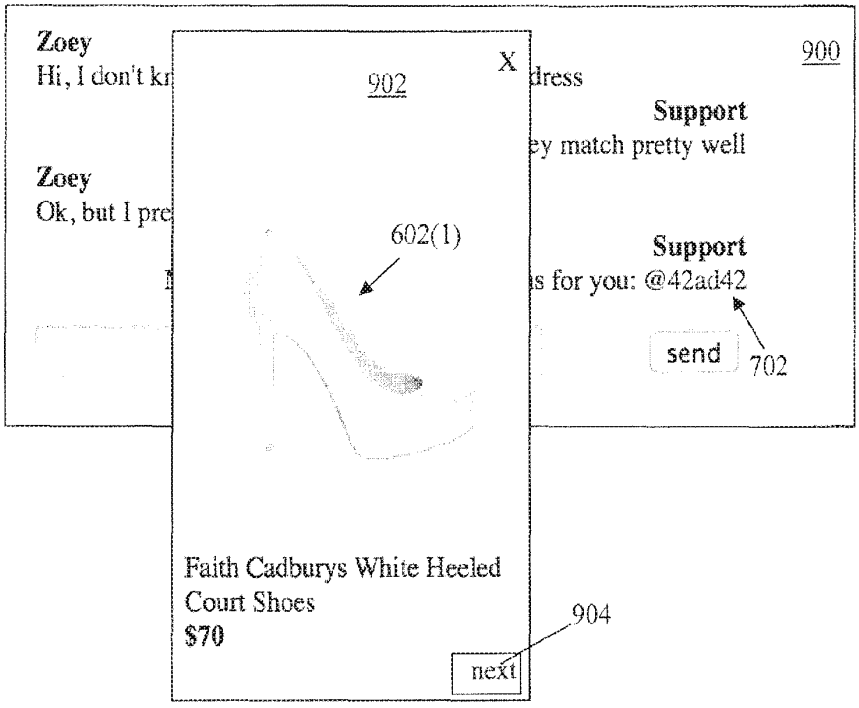


FIG. 9

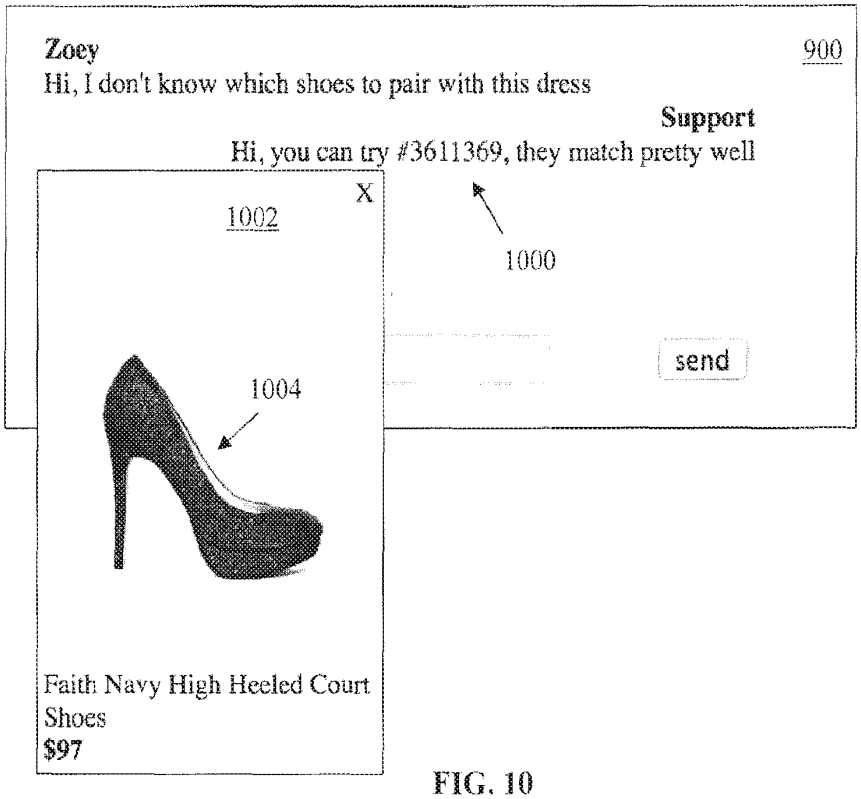


FIG. 10

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 15/36956

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - G06F 17/30 (2015.01)
CPC - G06F 17/30
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC(8) Classification(s): G06F 17/30, 13/00; G06Q 30/00 (2015.01)
CPC Classification(s): G06F 17/30, 17/30864; Y10S 707/99933; G06Q 30/00; H04L 51/04

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

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
PatSeer (US, EP, WO, JP, DE, GB, CN, FR, KR, ES, AU, IN, CA, Other Countries (INPADOC), RU, AT, CH, TH, BR, PH); IEEE/IEEEXplore, Google/Google Scholar; IP.com; Keywords: web chat, live chat, text, SMS, panel, window, search, query, keyword, terms, token, symbol, character, icon, image, avatar, identifier, reference, results, response, preview, portion, segment, area, website, URI, URL, hyperlink

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 6678673 B1 (EVES, D et al.) January 13, 2004; abstract; column 3, lines 46-52; column 4, lines 44-55; column 7, lines 10-16; column 7, lines 20-27	1-5, 7-12, 14-19, 21
Y		6, 13, 20
Y	US 6804664 B1 (HARTMAN, B et al.) October 12, 2004, abstract	6, 13, 20
A	US 2013/031 1339 A1 (JEREMIAS, L) November 21, 2013; entire document	1-21

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

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

Date of the actual completion of the international search	Date of mailing of the international search report
28 August 2015 (28.08.2015)	16 SEP 2015
Name and mailing address of the ISA/ Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-8300	Authorized officer Shane Thomas PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

Form PCT/ISA/210 (second sheet) (January 2015)

Electronic Acknowledgement Receipt

EFS ID:	40273079
Application Number:	16430210
International Application Number:	
Confirmation Number:	6600
Title of Invention:	Modular Systems and Methods For Selectively Enabling Cloud-Based Assistive Technologies
First Named Inventor/Applicant Name:	Sean D. Bradley
Customer Number:	20995
Filer:	Xiaoyu Wang/Kealani Aguon
Filer Authorized By:	Xiaoyu Wang
Attorney Docket Number:	AUDEY.003C1
Receipt Date:	13-AUG-2020
Filing Date:	03-JUN-2019
Time Stamp:	13:44:52
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	Foreign Reference	WO_2001075678_A1.pdf	9891177 109dd550cba275b5e966417aaf9e00198a6a66d1	no	103

Warnings:

Information:					
2	Foreign Reference	WO_2009111251_A1.pdf	1119962 609de0fe404d298f81c8fd36cfc39d9c7bb621f7	no	29
Warnings:					
Information:					
3	Foreign Reference	WO_2011031868_A1.pdf	1242663 ca46e7b4cf0f6b4e7ed68d1bfa434d1aa737d447	no	31
Warnings:					
Information:					
4	Foreign Reference	WO_2011156739_A2.pdf	1143721 3512389d7ada7073e4e1d9d4d28b2b940741c7da	no	30
Warnings:					
Information:					
5	Foreign Reference	WO_2011156743_A2.pdf	1163155 eeed366efc8bb2f42d9ceb2acb9271075e2fc994	no	33
Warnings:					
Information:					
6	Foreign Reference	WO_2012024380_A3.pdf	299546 836dd7a5e068547616c0ef16b2958f31ef726cb3	no	3
Warnings:					
Information:					
7	Foreign Reference	WO_2012036833_A9.pdf	296374 2414cf933d8e3989f38d8b452349ec92628bbebd	no	3
Warnings:					
Information:					
8	Foreign Reference	WO_2012064856_A3.pdf	305978 ebedc6a31bc5708a739d55b7bfe6adce6a0a893e	no	3
Warnings:					
Information:					

9	Foreign Reference	WO_2012064857_A3.pdf	294365 90984b3256c80bd027522505fd34433b4b4d46fc	no	3
Warnings:					
Information:					
10	Foreign Reference	WO_2012088326_A9.pdf	362880 1b4a5c14880edba1398fa7f072119ad53b158ba8	no	5
Warnings:					
Information:					
11	Foreign Reference	WO_2012178167_A3.pdf	319896 3ca676e34b80664e55e78953d6b05661f163219b	no	4
Warnings:					
Information:					
12	Foreign Reference	WO_2013003455_A3.pdf	307670 effa1fc8275cf8b58a76d4a9aed4839a8a046670	no	3
Warnings:					
Information:					
13	Foreign Reference	WO_2013112285_A1.pdf	1336215 21c739d2720bbfefb4e805276ddd91c8333c59a2	no	33
Warnings:					
Information:					
14	Foreign Reference	WO_2014040080_A1.pdf	4616515 bb144793573c939fb31305f3bcfeb76b57c2aa64	no	42
Warnings:					
Information:					
15	Foreign Reference	WO_2015171228_A1.pdf	2072418 240484e475e85b6f05d5509b317462f8131c0527	no	41
Warnings:					
Information:					

16	Foreign Reference	WO_2016032602_A1.pdf	1901713 6e3918236339e6ae876b26bc65cfe7ad43bbaeec	no	37
Warnings:					
Information:					
17	Foreign Reference	WO_2016057092_A1.pdf	1460619 1ec4bb7ac0bfd0276c26197fea637952deb2f975	no	32
Warnings:					
Information:					
		Total Files Size (in bytes):	28134867		

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/08 Equivalent

INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Application No.	16/430210
	Filing Date	June 03, 2019
	First Named Inventor	Bradley, Sean D.
	Art Unit	2177
(Multiple sheets used when necessary)	Examiner	Zuberi, Mohammed H.
SHEET 1 OF 5	Attorney Docket No.	AUDEY.003C1

U.S. PATENT DOCUMENTS

Examiner Initials	Cite No.	Document Number *Number - Kind Code (if known)* Example: 1,234,567 B1	Publication Date MM-DD-YYYY	Name of Patentee or Applicant	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear
	1	8,321,502	11-27-2012	Scoda	
	2	8,516,362	08-20-2013	Scoda	
	3	8,527,862	09-03-2013	Scoda et al.	
	4	8,589,484	11-19-2013	Scoda	
	5	8,650,474	02-11-2014	Scoda	
	6	8,868,638	10-21-2014	Scoda	
	7	8,983,935	03-17-2015	Scoda	
	8	8,984,164	03-17-2015	Scoda	
	9	9,275,023	03-01-2016	Scoda	
	10	9,311,281	04-12-2016	Scoda	
	11	9,319,244	04-19-2016	Taylor et al.	
	12	9,473,592	10-18-2016	Scoda	
	13	9,547,633	01-17-2017	Scoda	
	14	9,736,245	08-15-2017	Scoda	
	15	9,846,572	12-19-2017	Scoda	
	16	9,846,686	12-19-2017	Scoda	
	17	9,876,776	01-23-2018	Scoda	
	18	9,971,636	05-15-2018	Scoda et al.	
	19	10,015,226	07-03-2018	Scoda et al.	
	20	10,049,089	08-14-2018	Scoda et al.	
	21	10,116,726	10-30-2018	Scoda	
	22	10,120,847	11-06-2018	Scoda	
	23	10,198,414	02-05-2019	Scoda	
	24	10,282,401	05-07-2019	Scoda	
	25	10,218,775	02-026-2019	Scoda	

Examiner Signature	Date Considered

***Examiner:** Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

T[1] - Place a check mark in this area when an English language Translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Application No.	16/430210
	Filing Date	June 03, 2019
	First Named Inventor	Bradley, Sean D.
	Art Unit	2177
(Multiple sheets used when necessary)	Examiner	Zuberi, Mohammed H.
SHEET 2 OF 5	Attorney Docket No.	AUDEY.003C1

U.S. PATENT DOCUMENTS

Examiner Initials	Cite No.	Document Number *Number - Kind Code (if known)* Example: 1,234,567 B1	Publication Date MM-DD-YYYY	Name of Patentee or Applicant	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear
	26	10,423,709 (Our Ref. AUDEY.002A), including its prosecution history, the cited references, and the Office Actions therein	09-24-2019	Bradley et al.	
	27	10,444,934 (Our Ref. AUDEY.003A), including its prosecution history, the cited references, and the Office Actions therein	10-15-2019	Bradley et al.	
	28	10,452,730	10-22-2019	Scoda	
	29	2011/0307855	12-15-2011	Scoda et al.	
	30	2013/0305139	11-14-2013	Scoda	
	31	2014/0149447	05-29-2014	Scoda	
	32	2015/0319215	11-05-2015	Scoda	
	33	2017/0111431	04-20-2017	Scoda et al.	
	34	2018/0191764	07-05-2018	Chawla et al.	
	35	2018/0239516	08-23-2018	Scoda	
	36	2018/0349004	12-06-2018	Scoda et al.	

FOREIGN PATENT DOCUMENTS

Examiner Initials	Cite No.	Foreign Patent Document *Country Code-Number-Kind Code* Example: JP 1234567 A1	Publication Date MM-DD-YYYY	Name of Patentee or Applicant	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear	T[1]
	37	CA2716635C	11-01-2016	USABLENET INC.		
	38	CA2732540C	02-27-2018	USABLENET INC.		
	39	CA2773088A1	03-17-2011	USABLENET INC.		
	40	CA2800723A1	12-15-2011	USABLENET INC.		
	41	CA2800790A1	12-15-2011	USABLENET INC.		
	42	CA2807320A1	02-23-2012	USABLENET, INC.		

Examiner Signature	Date Considered

***Examiner:** Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

T[1] - Place a check mark in this area when an English language Translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Application No.	16/430210
	Filing Date	June 03, 2019
	First Named Inventor	Bradley, Sean D.
	Art Unit	2177
(Multiple sheets used when necessary)	Examiner	Zuberi, Mohammed H.
SHEET 3 OF 5	Attorney Docket No.	AUDEY.003C1

FOREIGN PATENT DOCUMENTS

Examiner Initials	Cite No.	Foreign Patent Document *Country Code-Number-Kind Code* Example: JP 1234567 A1	Publication Date MM-DD-YYYY	Name of Patentee or Applicant	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear	T[1]
	43	CA2811207A1	03-22-2012	USABLENET, INC.		
	44	CA2816336A1	05-18-2012	USABLENET INC.		
	45	CA2816338A1	05-18-2012	USABLENET INC.		
	46	CA2821769A1	06-28-2012	USABLENET INC.		
	47	CA2824861A1	03-10-2014	USABLENET INC.		
	48	CA2834466A1	05-29-2014	USABLENET INC.		
	49	CA2839006C	08-01-2017	USABLENET INC.		
	50	CA2839013A1	01-03-2013	USABLENET INC.		
	51	CA2843938A1	09-13-2014	USABLENET INC.		
	52	CA2845279A1	09-13-2014	USABLENET INC.		
	53	CA2848175A1	10-03-2014	USABLENET INC.		
	54	CA2855420A1	01-09-2015	USABLENET INC.		
	55	CA2858590A1	02-28-2015	USABLENET INC.		
	56	CA2861602A1	08-01-2013	USABLENET INC.		
	57	CA2868162A1	05-04-2015	USABLENET INC.		
	58	CA2944659A1	04-15-2017	USABLENET INC.		
	59	CA2947402A1	11-12-2015	USABLENET INC.		
	60	CA2951659A1	06-22-2017	USABLENET INC.		
	61	CA2958235A1	03-03-2016	USABLENET INC.		
	62	CA2963393A1	04-14-2016	USABLENET INC.		
	63	EP2363995B1	07-29-2015	USABLENET INC.		
	64	EP2476063A4	02-21-2018	USABLENET INC.		
	65	EP2580686A4	11-30-2016	USABLENET INC.		
	66	EP2580699A4	07-27-2016	USABLENET INC.		
	67	EP2606436A4	11-15-2017	USABLENET INC.		
	68	EP2616962A4	01-17-2018	USABLENET INC.		
	69	EP2638681B1	07-19-2017	USABLENET INC.		

Examiner Signature	Date Considered
***Examiner:** Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.	

T[1] - Place a check mark in this area when an English language Translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Application No.	16/430210
	Filing Date	June 03, 2019
	First Named Inventor	Bradley, Sean D.
	Art Unit	2177
(Multiple sheets used when necessary)	Examiner	Zuberi, Mohammed H.
SHEET 4 OF 5	Attorney Docket No.	AUDEY.003C1

FOREIGN PATENT DOCUMENTS

Examiner Initials	Cite No.	Foreign Patent Document *Country Code-Number-Kind Code* Example: JP 1234567 A1	Publication Date MM-DD-YYYY	Name of Patentee or Applicant	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear	T[1]
	70	EP2638683B1	03-15-2017	USABLENET INC.		
	71	EP2656303A4	07-30-2014	USABLENET INC.		
	72	EP2724251B1	08-08-2018	USABLENET INC.		
	73	EP2724253A4	04-20-2016	USABLENET INC.		
	74	EP2732385A4	04-29-2015	USABLENET INC.		
	75	EP2738696A1	06-04-2014	USABLENET INC.		
	76	EP2778975A1	09-17-2014	USABLENET INC.		
	77	EP2778976B1	12-27-2017	USABLENET INC.		
	78	EP2787454A1	10-08-2014	USABLENET INC.		
	79	EP2807578A4	11-18-2015	USABLENET INC.		
	80	EP2827261A1	01-21-2015	USABLENET INC.		
	81	EP2851813A1	03-25-2015	USABLENET INC.		
	82	EP2869222A1	05-06-2015	USABLENET INC.		
	83	EP3140733A4	12-13-2017	Usablenet Inc.		
	84	EP3156919A1	04-19-2017	USABLENET INC.		
	85	EP3185145A1	06-28-2017	USABLENET INC.		
	86	EP3195136A4	04-25-2018	USABLENET INC.		
	87	EP3204874A4	07-04-2018	USABLENET INC.		
	88	WO 2001/75678	10-11-2001	USABLENET, INC.		
	89	WO2009111251A1	09-11-2009	USABLENET INC.		
	90	WO2011031868A1	03-17-2011	USABLENET INC.		
	91	WO2011156739A2	12-15-2011	USABLENET INC.		
	92	WO2011156743A2	12-15-2011	USABLENET INC.		
	93	WO2012024380A3	06-14-2012	USABLENET, INC.		
	94	WO2012036833A9	12-20-2012	USABLENET, INC.		
	95	WO2012064856A3	10-04-2012	USABLENET INC.		
	96	WO2012064857A3	09-27-2012	USABLENET INC.		

Examiner Signature	Date Considered
***Examiner:** Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.	

T[1] - Place a check mark in this area when an English language Translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT	Application No.	16/430210
	Filing Date	June 03, 2019
	First Named Inventor	Bradley, Sean D.
	Art Unit	2177
(Multiple sheets used when necessary)	Examiner	Zuberi, Mohammed H.
SHEET 5 OF 5	Attorney Docket No.	AUDEY.003C1

FOREIGN PATENT DOCUMENTS						
Examiner Initials	Cite No.	Foreign Patent Document *Country Code-Number-Kind Code* Example: JP 1234567 A1	Publication Date MM-DD-YYYY	Name of Patentee or Applicant	Pages, Columns, Lines Where Relevant Passages or Relevant Figures Appear	T[1]
	97	WO2012088326A9	09-20-2012	USABLENET INC.		
	98	WO2012178167A3	02-28-2013	USABLENET INC.		
	99	WO2013003455A3	03-14-2013	USABLENET INC.		
	100	WO2013112285A1	08-01-2013	USABLENET INC.		
	101	WO2014040080A1	03-13-2014	USABLENET INC.		
	102	WO2015171228A1	11-12-2015	USABLENET INC.		
	103	WO2016032602A1	03-03-2016	USABLENET INC.		
	104	WO2016057092A1	04-14-2016	USABLENET INC.		

NON PATENT LITERATURE DOCUMENTS			
Examiner Initials	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	T[1]
	105		

33320834

Examiner Signature	Date Considered
***Examiner:** Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.	

T[1] - Place a check mark in this area when an English language Translation is attached.

Electronic Acknowledgement Receipt

EFS ID:	40272248
Application Number:	16430210
International Application Number:	
Confirmation Number:	6600
Title of Invention:	Modular Systems and Methods For Selectively Enabling Cloud-Based Assistive Technologies
First Named Inventor/Applicant Name:	Sean D. Bradley
Customer Number:	20995
Filer:	Xiaoyu Wang/Kealani Aguon
Filer Authorized By:	Xiaoyu Wang
Attorney Docket Number:	AUDEY.003C1
Receipt Date:	13-AUG-2020
Filing Date:	03-JUN-2019
Time Stamp:	13:43:37
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	Foreign Reference	CA_2716635_C.pdf	2892924 055aff219ae60fbe2c8b641bdddd070bb85573ee	no	30

Warnings:

Information:					
2	Foreign Reference	CA_2732540_C.pdf	2871804 1c21ddd9238fa5cd4defd2c7f2b254e29adf2758	no	28
Warnings:					
Information:					
3	Foreign Reference	CA_2773088_A1.pdf	3134787 e74ad7b5d9db364336b69b1b24fbd1cd485dd52a	no	33
Warnings:					
Information:					
4	Foreign Reference	CA_2800723_A1.pdf	3523417 5b4771254cbee145790d9a20c31555cf194649a9	no	35
Warnings:					
Information:					
5	Foreign Reference	CA_2800790_A1.pdf	3065359 611a390368ef310bdcb5aa5229c712937d418c0f	no	33
Warnings:					
Information:					
6	Foreign Reference	CA_2807320_A1.pdf	2863642 7c02fb982d6bdbcee573fb26f5d3e854a66c78ed	no	28
Warnings:					
Information:					
7	Foreign Reference	CA_2811207_A1.pdf	2712330 dcfb957835fcad52044e6734f80621a9bdc7b5cf	no	25
Warnings:					
Information:					
8	Foreign Reference	CA_2816336_A1.pdf	3505903 d9d62f3e0a05bda3994cbc978116b4f07eedfaf0	no	30
Warnings:					
Information:					

9	Foreign Reference	CA_2816338_A1.pdf	3766501 40c7e4e797f3cab25af31fbd13b2bc6ab8f2246f	no	35
Warnings:					
Information:					
10	Foreign Reference	CA_2821769_A1.pdf	1967079 c537491bfa40eb0d978302ef156c360efde6b0f0	no	22
Warnings:					
Information:					
11	Foreign Reference	CA_2824861_A1.pdf	5306419 e731db4447228b5b41ab3df2f414c6c67c9f655f	no	42
Warnings:					
Information:					
12	Foreign Reference	CA_2834466_A1.pdf	2483968 d01c06decb2b87deead6275807da33169403029f	no	24
Warnings:					
Information:					
13	Foreign Reference	CA_2839006_C.pdf	2749415 a12b6cb8950a8ddda901bf4dfb6da497ea557d9b	no	25
Warnings:					
Information:					
14	Foreign Reference	CA_2839013_A1.pdf	2837468 2385f5fde50b8c0161d784d997d9c6aba2c5cb3a	no	27
Warnings:					
Information:					
15	Foreign Reference	CA_2843938_A1.pdf	2633080 5a642678eebca332fb57a70280c8f76f40c63989	no	26
Warnings:					
Information:					

16	Foreign Reference	CA_2845279_A1.pdf	3702167 cb34b819c2fd878aad5f59622aebf9d9456cff60	no	31
Warnings:					
Information:					
17	Foreign Reference	CA_2848175_A1.pdf	4358328 41e2aa268298b978f554af9be8b7fa5bd6cfea0a	no	37
Warnings:					
Information:					
18	Foreign Reference	CA_2855420_A1.pdf	3335159 2a191cfbc8641b4e9e03d026f9bb48721e6aa296	no	29
Warnings:					
Information:					
19	Foreign Reference	CA_2858590_A1.pdf	3458795 4032f9ee84ee42e7e62e4d493c4050b103d57dfe	no	30
Warnings:					
Information:					
20	Foreign Reference	CA_2861602_A1.pdf	3676168 cdcdcb161cfcfc233447fe654d0fbc3c1d473b6d	no	33
Warnings:					
Information:					
21	Foreign Reference	CA_2868162_A1.pdf	4402882 4098b1066d8d3fa1c40b048805fd0f6fd2073f16	no	39
Warnings:					
Information:					
22	Foreign Reference	CA_2944659_A1.pdf	2484610 f992f49179d5c690e1ef889db5b54bd517db67dc	no	27
Warnings:					
Information:					

23	Foreign Reference	CA_2947402_A1.pdf	4508928 b29ff35f8d88f32a8bba39dd9c18b333c9fab2cd	no	42
Warnings:					
Information:					
24	Foreign Reference	CA_2951659_A1.pdf	4204943 f0fe8d027ac69ca83c23e975621376c508b8eed1	no	36
Warnings:					
Information:					
25	Foreign Reference	CA_2958235_A1.pdf	5202083 8569c73193917a7d82f4c6f136f404dee254c134	no	39
Warnings:					
Information:					
26	Foreign Reference	CA_2963393_A1.pdf	4428398 5d455c7517d613b6deb5deb71af61ce14bb9b75a	no	34
Warnings:					
Information:					
27	Foreign Reference	EP_2363995_B1.pdf	440776 fcbe03df9c9da1b0e3a1d9bff470b5fc8a78223a	no	15
Warnings:					
Information:					
28	Foreign Reference	EP_2476063_A4.pdf	91506 509673906f8cddb35ffdea8cd6924bc9f90ab1b2	no	1
Warnings:					
Information:					
29	Foreign Reference	EP_2580686_A4.pdf	2152794 af1b6808c8b6911c3c83754863b0918a20bbeda8	no	34
Warnings:					
Information:					

30	Foreign Reference	EP_2580699_A4.pdf	1813058 93654570e3deafc72fa0668b24dbe5c915d41761	no	31
Warnings:					
Information:					
31	Foreign Reference	EP_2606436_A4.pdf	1721326 208f48aaf1466d8c658c5e05ff86f2bd2dd7f743	no	27
Warnings:					
Information:					
32	Foreign Reference	EP_2616962_A4.pdf	1704053 3e759fc7b148f02be08d8d243f8a2f31f3145c0e	no	25
Warnings:					
Information:					
33	Foreign Reference	EP_2638681_B1.pdf	522325 274e00aa99fd46d155e44a03a3fa1230875f18b3	no	16
Warnings:					
Information:					
34	Foreign Reference	EP_2638683_B1.pdf	461331 378444bed48f5525eab696322ffc6ef417f41896	no	21
Warnings:					
Information:					
35	Foreign Reference	EP_2656303_A4.pdf	1311043 0f5b007e5be06019cc333f03bb5c4bb7e540d0dc	no	23
Warnings:					
Information:					
36	Foreign Reference	EP_2724251_B1.pdf	283696 d1ba6d8d16c3f8911e49d98d043b053d11c96f6a	no	12
Warnings:					
Information:					

37	Foreign Reference	EP_2724253_A4.pdf	1583611 43101b2a73ee9b70f009a2c5576e9b73887a5ba7	no	26
Warnings:					
Information:					
38	Foreign Reference	EP_2732385_A4.pdf	2919919 b900b6f6c7b2a64734592e11b882cfa17c2171c0	no	43
Warnings:					
Information:					
39	Foreign Reference	EP_2738696_A1.pdf	641949 f996f62d12541e50be782ee3cb1a0b04a8fea9a1	no	12
Warnings:					
Information:					
40	Foreign Reference	EP_2778975_A1.pdf	398309 914806e1c60119a50839731c3489c1b53e78d257	no	15
Warnings:					
Information:					
41	Foreign Reference	EP_2778976_B1.pdf	388760 6691ec5a997c4e053442996b4d0a941d48fdc244	no	17
Warnings:					
Information:					
42	Foreign Reference	EP_2787454_A1.pdf	567534 2350bfbb89d5f9a7db7cf613b08123c649529840	no	16
Warnings:					
Information:					
43	Foreign Reference	EP_2807578_A4.pdf	2182742 f604de720e908cc356b8049c6b22156b1dec78f7	no	34
Warnings:					
Information:					

44	Foreign Reference	EP_2827261_A1.pdf	438940 1b404e0f7e477f4db8a742e30492198cee1c8293	no	14
Warnings:					
Information:					
45	Foreign Reference	EP_2851813_A1.pdf	486783 6ed87618dee8f0c583155085880c229320ec6c44	no	16
Warnings:					
Information:					
46	Foreign Reference	EP_2869222_A1.pdf	754467 6f67d2f78f02192cb3cec4fd5b8c4edf227f5fc4	no	21
Warnings:					
Information:					
47	Foreign Reference	EP_3140733_A4.pdf	2963802 c54295b5dc1ba01e47f5385f532ed74fbe0fd0d7	no	42
Warnings:					
Information:					
48	Foreign Reference	EP_3156919_A1.pdf	430138 f4a6941c9c949678fc9309691927e7db5179f78f	no	16
Warnings:					
Information:					
49	Foreign Reference	EP_3185145_A1.pdf	760593 c50932c2ee63a79969b75f8bae62cb8b50f0aacd	no	21
Warnings:					
Information:					
50	Foreign Reference	EP_3195136_A4.pdf	3125237 78a92160334a48f59fe7d8246ecf4e09eac24303	no	38
Warnings:					
Information:					

51	Foreign Reference	EP_3204874_A4.pdf	2573516 bd0c2a18b7d7e40d136b8518baa141f2c3de1d41	no	33
Warnings:					
Information:					
52		IDS.pdf	69912 bbb9a3b293eec32a6e7a548b7c07543c7a80a8df	yes	7

	Multipart Description/PDF files in .zip description		
	Document Description	**Start**	**End**
	Transmittal Letter	1	2
	Information Disclosure Statement (IDS) Form (SB08)	3	7

Warnings:	
Information:	
Total Files Size (in bytes):	120864677

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.

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2716635 C 2016/11/01

(11)(21) 2 716 635

(12) BREVET CANADIEN
CANADIAN PATENT

(13) C

(86) Date de dépôt PCT/PCT Filing Date: 2009/02/26
(87) Date publication PCT/PCT Publication Date: 2009/09/11
(45) Date de délivrance/Issue Date: 2016/11/01
(85) Entrée phase nationale/National Entry: 2010/08/23
(86) N° demande PCT/PCT Application No.: US 2009/035206
(87) N° publication PCT/PCT Publication No.: 2009/111251
(30) Priorité/Priority: 2008/02/29 (US12/072,987)

(51) Cl.Int./Int.Cl. *G06F 17/00* (2006.01), *G06F 3/0484* (2013.01), *G06F 3/14* (2006.01)

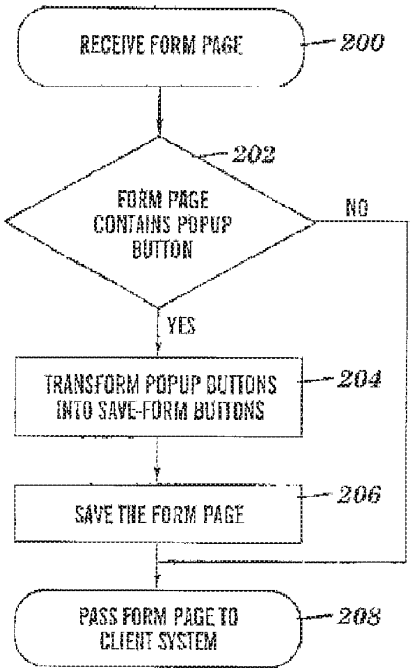
(72) Inventeur/Inventor:
SCODA, ENRICO, IT

(73) Propriétaire/Owner:
USABLENET INC., US

(74) Agent: GOWLING WLG (CANADA) LLP

(54) Titre : PROCEDES POUR UNE OPTIMISATION DE CONTENU INTERNET DANS DES ENVIRONNEMENTS D'AFFICHAGE A PAGE UNIQUE ET SYSTEMES APPARENTES

(54) Title: METHODS FOR WEB CONTENT OPTIMIZATION IN SINGLE PAGE DISPLAY ENVIRONMENTS AND SYSTEMS THEREOF



(57) Abrégé/Abstract:
A method, computer readable medium and system for web content optimization in single page display environment includes receiving in a system with a single page display environment an engagement of an activator for a data picker for at least one form field in a form page. The form page is replaced with a data picker web page associated with the engaged activator in the single page display environment. A selection for the at least one form field from the data picker web page is received. The data picker web page is replaced with the form page in the single page display environment. The form page is filled with any previously entered data and the at least one form field with the received selection.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
11 September 2009 (11.09.2009)

PCT

(10) International Publication Number
WO 2009/111251 A1

(51) International Patent Classification:
G06F 17/21 (2006.01)

(21) International Application Number:
PCT/US2009/035206

(22) International Filing Date:
26 February 2009 (26.02.2009)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/072,987 29 February 2008 (29.02.2008) US

(71) Applicant *(for all designated States except US)*: **US-ABLENET INC.** [US/US]; 101 Avenue Of The Americas, Suite 1204, New York, NY 10013 (US).

(72) Inventor: **SCODA, Enrico**; Via Cividina 416/3, I-33035 Martignacco (UD) (IT).

(74) Agents: **LEINBERG, Gunnar, G.** et al.; Nixon Peabody LLP, 1100 Clinton Square, Rochester, NY 14604-1792 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(54) Title: METHODS FOR WEB CONTENT OPTIMIZATION IN SINGLE PAGE DISPLAY ENVIRONMENTS AND SYSTEMS THEREOF

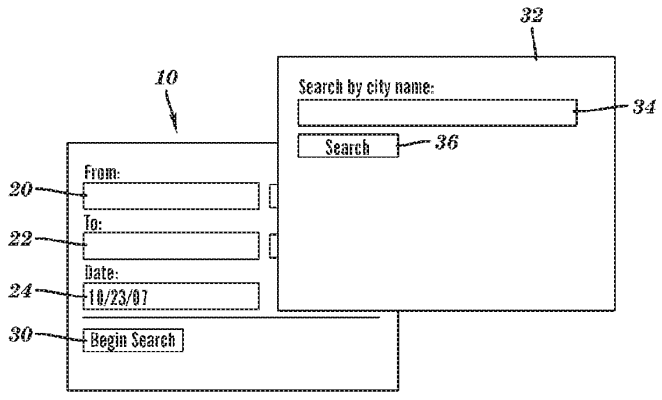


FIG. 1B

(57) Abstract: A method, computer readable medium and system for web content optimization in single page display environment includes receiving in a system with a single page display environment an engagement of an activator for a data picker for at least one form field in a form page. The form page is replaced with a data picker web page associated with the engaged activator in the single page display environment. A selection for the at least one form field from the data picker web page is received. The data picker web page is replaced with the form page in the single page display environment. The form page is filled with any previously entered data and the at least one form field with the received selection.

WO 2009/111251 A1

METHODS FOR WEB CONTENT OPTIMIZATION IN SINGLE PAGE DISPLAY ENVIRONMENTS AND SYSTEMS THEREOF

FIELD OF THE INVENTION

[001] The present invention relates to a methods and systems for web content optimization and, more particularly, to methods for web content optimization in single page display environments and systems thereof.

BACKGROUND

[002] Many web sites, such as air, train, or car booking sites, use popup windows or data pickers to enhance the web page interface and make selection of data such as dates or locations, easier for the user.

[003] Referring to FIGS. 1A-1D, an example of a popup window 32 used to find and select airport codes in a web page window 10 for booking a flight on a given date is illustrated. As shown in FIG. 1A, the original web page window 10 includes a "From" field 20 to specify the departure location of the flight, a "To" field 22 to specify the arrival destination of the flight, and a "Date" field 24 to specify the departure date of the flight. Additionally, "Find Airport" buttons 26 and 28 are located adjacent "From" field 20 and "To" field 22, respectively, and a "Begin Search" button 30 is located on web page window 10 as well. If a user does not know the code of the departure or arrival airport, the user can press the corresponding one of the "Find Airport" buttons 26 and 28 to search for the airport code.

[004] As shown in FIG. 1B, if either of the "Find Airport" buttons 26 and 28 is engaged, a new popup window 32 with a "Search by city name" field 34 and a "Search" button 36 appears. The user can type in the name of the city in the "Search by city name" field 34 and press the "Search" button 36 to get one or more codes of airport near the entered city, such as "JFK/John F Kennedy International (JFK)" text field 38 and a "LGA/La Guardia (LGA)" text field 40, as shown in FIG. 1C.

[005] When the "JFK/John F Kennedy International (JFK)" text field 38 in the popup window 32 is selected, the code is copied in the "From" field 20 as

shown in FIG. 1D and the popup window 32 is closed. These operations of opening the popup window and copying the result value to the original field can only be performed by script enabled web browsers that can open more than one web page at the same time.

[006] Unfortunately, many devices, such as most mobile phones, smart phones, and personal digital assistants (PDAs), do not allow popup windows to be opened because their screen size is too small. Additionally, some desktop web browsers are configured to block popup windows. Further, the scripting capabilities of these devices are often very limited resulting in need for alternative solutions.

SUMMARY

[007] A method for web content optimization in single page display environment in accordance with embodiments of the present invention includes receiving in a system with a single page display environment an engagement of an activator for a data picker for at least one form field in a form page. The form page is replaced with a data picker web page associated with the engaged activator in the single page display environment. A selection for the at least one form field from the data picker web page is received. The data picker web page is replaced with the form page in the single page display environment. The form page is filled with any previously entered data and the at least one form field with the received selection.

[008] A computer readable medium in accordance with other embodiments of the present invention includes having stored thereon instructions for web content optimization in single page display environment comprising machine executable code which when executed by at least one processor, causes the processor to perform steps comprising receiving in a system with a single page display environment an engagement of an activator for a data picker for at least one form field in a form page. The form page is replaced with a data picker web page associated with the engaged activator in the single page display environment. A selection for the at least one form field from the data picker web page is received. The data picker web page is replaced with the form page in the single

page display environment. The form page is filled with any previously entered data and the at least one form field with the received selection.

[009] A system with web content optimization in single page display environment in accordance with other embodiments of the present invention includes a display system with a single page display format and a web page processing system. The web processing system receives an engagement of an activator for a data picker for at least one form field in a form page on display on the display system. The web processing system replaces the form page with a data picker web page associated with the engaged activator and receives a selection for the at least one form field from the data picker web page. The web processing system replaces the data picker web page with the form page in the single page display environment and fills the form page with any previously entered data and the at least one form field with the received selection.

[0010] The present invention provides web optimization systems and methods which enable the functionality of popup data pickers inside web applications to be executed in display environments which only permit viewing one page at a time. With the present invention, a much larger number of web applications which require the functionality of one or more popup windows can be viewed and used in devices with single page display format, such as mobile phones, smart phones, and PDAs or in desktop web browsers or other systems configured to block popup windows.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] FIGS. 1A-1D are prior art diagrams of a booking page window for an airline web application with a popup window to find and select an airport code;

[0012] FIG. 2 is a block diagram of a system which optimizes web content in a device with a single display page environment in accordance with embodiments of the present invention;

[0013] FIGS. 3A-3D are diagrams of a booking page window for an airline web application and a popup window to find and select an airport code in a single

page display environment in accordance with embodiments of the present invention;

[0014] FIG. 4 is a flow chart of a method for processing a form page containing a popup window;

[0015] FIG. 5 is a flow chart of a method for handling a request for a popup window; and

[0016] FIG. 6 is a flow chart of a method filling a form page with values obtained with a popup window.

DETAILED DESCRIPTION

[0017] A web content optimization system 100 in accordance with embodiments of the present invention is illustrated in FIG. 2. The web content optimization system 100 is coupled to one or more client systems 102(1)-102(n) and a web server system 104, although web content optimization system 100 could be connected to other types and numbers of servers, systems, devices, and components and in other manners. The present invention provides a number of advantages including providing web optimization systems and methods which enable the functionality of a popup window inside a web application to be executed in display environments which only permit viewing one page at a time.

[0018] Referring more specifically to FIG. 1, the web content optimization system 100 enables the functionality of a popup window inside a web application to be executed in display environments which only permit viewing one page at a time, although the web content optimization system 100 can provide other numbers and types of functions. Although one web content optimization system 100 is shown, other numbers and types of web content optimization systems can be used.

[0019] The web content optimization system 100 includes a central processing unit (CPU) or processor, a memory, and an interface system which are coupled together by a bus or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and

locations can be used. The processor in the web content optimization system 100 executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[0020] The memory in the web content optimization system 100 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor, can be used for the memory in the web content optimization system 100.

[0021] The interface system in the web content optimization system 100 is used to operatively couple and communicate between the web content optimization system 100 and the client systems 102(1)-102(n) and the web server system 104 via the Internet, although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[0022] Each of the client systems 102(1)-102(n) enables a user to access content and utilize one or more applications from the web server system 104 through the web content optimization system 100 through one or more communication networks, although one or more of the client systems 102(1)-102(n) could access content and utilize other types and numbers of applications from other sources and could provide a wide variety of other functions for the user. Although multiple client systems 102(1)-102(n) are shown, other numbers and types of user computing systems could be used. In this example, the client systems 102(1)-102(n) comprise mobile devices with Internet access that only permit a single web page to be displayed, although each of the client systems 102(1)-102(n) can comprise a wide variety of different types of devices and

systems, such as smart phones, PDAs, or desktop computers or other systems configured to block popup windows.

[0023] Each of the client systems 102(1)-102(n) includes a central processing unit (CPU) or processor, a memory, user input device, a display with a single page display environment, and an interface system, and which are coupled together by a bus or other link, although one or more of the client systems can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor in each of the client systems 102(1)-102(n) executes a program of stored instructions for one or more aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions.

[0024] The memory in each of the client systems 102(1)-102(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to one or more processors, can be used for the memory in each of the client systems 102(1)-102(n).

[0025] The user input device in each of the client systems 102(1)-102(n) is used to input selections, such as requests for an application, although the user input device could be used to input other types of data and interact with other elements. The user input device can include a computer keyboard and a computer mouse, although other types and numbers of user input devices can be used. Input devices can include phone keypads, touch screens and voice inputs.

[0026] The display in each of the client systems 102(1)-102(n) is used to show data and information to the user, such as a travel booking web page by way of example only. The display in each of the client systems 102(1)-102(n) is a

CRT or LCD screen which only permits a single page to be displayed at a time, although other types and numbers of displays could be used.

[0027] The interface system in each of the client systems 102(1)-102(n) is used to operatively couple and communicate between the client systems 102(1)-102(n) and the web content optimization system 100 and web server system 104 over the Internet, although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[0028] The web server system 104 provides one or more web software applications for use by one or more of the client systems 102(1)-102(n), although the web server system 104 can provide other numbers and types of applications and/or content and can have provide other numbers and types of functions. Although one web server system 104 is shown, other numbers and types of web server systems can be used.

[0029] The web server system 104 includes a central processing unit (CPU) or processor, a memory, and an interface system which are coupled together by a bus or other link, although the web server system 104 could have other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor in the web server system 104 executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, including managing application functionality, although the processor could execute other numbers and types of programmed instructions.

[0030] The memory in the web server system 104 stores these programmed instructions for one or more aspects of the present invention as described and illustrated by way of the embodiments herein including managing application functionality, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic,

optical, or other reading and/or writing system that is coupled to the processor, can be used for the memory in each of the web server system 104.

[0031] The interface system in the web server system 104 is used to operatively couple and communicate between the web server system 104 and the web content optimization system 100 and the client systems 102(1)-102(n) via the Internet, although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[0032] Although embodiments of the web content optimization system 100, the client systems 102(1)-102(n), and the web server system 104, are described and illustrated herein, each of the client systems 102(1)-102(n), the web content optimization system 100, and the web server system 104, can be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s).

[0033] Furthermore, each of the systems of the embodiments may be conveniently implemented using one or more general purpose computer systems, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[0034] In addition, two or more computing systems or devices can be substituted for any one of the systems in any embodiment of the embodiments. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the embodiments. The embodiments may also be implemented on computer system or systems that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless

communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[0035] The embodiments may also be embodied as a computer readable medium having instructions stored thereon for one or more aspects of the present invention as described and illustrated by way of the embodiments herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the embodiments, as described and illustrated herein.

[0036] By way of example only, the operation of the web content optimization system 100 to process and manage the use of an airline web application from the web server system 104 on a single page display environment on one of the client systems 102(1)-102(n) in accordance with embodiments of the present invention will now be described with reference to FIGS. 3A-6.

[0037] Referring to FIGS. 3A and 4, in this particular example one of the client systems 102(1)-102(n) enters a Hyper Text Transfer Protocol (HTTP) request that specifies a Uniform Resource Locator (URL) for an airline booking web page 100 which is transmitted over the Internet to the web server system 104 via the web content optimization system 110, although other types of requests, other manners for transmitting the request, and other types of web pages can be retrieved. In step 200, the web server system 104 receives the request and transmits the requested airline booking web page 110 to the web content optimization system 100.

[0038] In step 202, the web content optimization system 100 determines if the received airline booking web page 110 has any buttons or other activators for one or more popup windows. If the web content optimization system 100 determines the requested web page does not contain one or more popup buttons or other activators for one or more popup windows, then the No branch is taken to step 208. If the web content optimization system 100 determines the requested

web page does contain one or more popup buttons or other activators for one or more popup windows, then the Yes branch is taken to step 204. In this particular example, the airline booking web page 110 has two "Find Airport" buttons 112 and 114 for popup windows to search for airport codes so the Yes branch is taken to step 204.

[0039] In step 204, the web content optimization system 100 converts the associated function of the popup buttons or other activators for one or more popup windows to buttons that activate the saving of the current web page along with any values entered by a user in one or fields of the current web page. These popup buttons are standard HTML (Hyper Text Markup Language) form submit buttons having the name attribute containing all the necessary information to substitute the current page with the desired popup page, although other types of popup buttons can be used and converted. In this particular example, the web content optimization system 100 converts the "Find Airport" buttons 112 and 114 from trying to retrieve and displaying a popup window to search for an airport code to buttons that activate the saving of the current airline booking web page 110 along with any values entered by a user in one or fields. Additionally, the buttons 112 and 114 are revised to activate retrieving and replacing the airline booking web page 110 with a page which has the popup window for searching for an airport code.

[0040] In step 206, the web content optimization system 100 saves the revised web page with the revised button or buttons. In this particular example, the web content optimization system 100 saves the revised airline booking webpage 110 with the converted "Find Airport" buttons 112 and 114.

[0041] By way of example only, a formal definition of the name attribute and two HTML fragments corresponding to two "Find Airport" save-form buttons is illustrated below:

<name_attribute> ::=
un_jtt_save_form/<redirect_url><form_name><field1_to_fill>…<fieldn_to_fill>

<input value="Find Airport" name="un_jtt_save_form/http%3A%2F%www.acme.com%2Ffindairport.jsp+form1+from" type="submit" />

<input value="Find Airport" name="un_jtt_save_form/http%3A%2F%www.acme.com%2Ffindairport.jsp+form1+to" type="submit" />

[0042] The attribute name contains all the necessary information to open the popup window correctly including: (1) the popup window page URL to redirect to after saving the form web page; (2) the name or index of the form web page; and (3) the list of fields in the form web page to be filled when a popup window is "clicked on" or otherwise activated.

[0043] In step 208, the revised form web page is transmitted to the display with the single web page display environment of the originally requesting one of the client system 102(1)-102(n). In this particular example, the revised airline booking web page 110 is displayed which includes the revised buttons 112 and 114 and also includes a "From" field 118, a "To" field 120, and a "Date" field 122 in which values can be entered by the user, although the displayed page can have other numbers and types of fields, buttons, and other content, depending on the particular application.

[0044] Referring to FIGS. 3B and 5, in step 210 if a user at the originally requesting one of the client systems 102(1)-102(n) needs additional information to complete one of the fields, then the user can press one of the revised buttons 112 or 114 which transmits a request for an associated popup window to the web content optimization system 100. In this particular example, if the user at one of the client systems 102(1)-102(n) does not know the code of the departure airport for the "From" field 118, then the user can press the corresponding "Find Airport" button 26 which transmits a request for the associated popup window to the web content optimization system 100.

[0045] In step 212, the web content optimization system 100 determines if the request for the popup window is associated with a revised web page that has new functionality associated with the activation of the button. If the request for the popup window is not associated with a revised web page that has new functionality associated with the activation of the button, then the No branch is taken to step 216. If the request for the popup window is associated with a revised web page that has new functionality associated with the activation of the button, then the Yes branch is taken to step 214. In this particular example, when the "Find Airport" button 26 is pressed, the request for the popup window is associated with a revised airline booking web page 110 that has new functionality associated with the activation of the button so the Yes branch is taken to step 214.

[0046] In step 214, the web content optimization system 110 saves the form web page along with any values entered in fields by a user, although other types and amounts of information can be saved in other manners. In this particular example, the airline booking web page 110 along with the value "10/23/07" entered by the user in the "Date" field 122 is saved, although other types and amounts of information can be saved.

[0047] In step 216, the web content optimization system 100 retrieves the requested popup window from the web server system 104 and transmits the popup window to the requesting one of the user client systems 102(1)-102(n). The requesting one of the user client systems 102(1)-102(n) only displays the new popup window in the screen of the display. In this particular example, the popup window 130 to assist with a search of an airport code is only shown on the display in the requesting one of the user client systems 102(1)-102(n) as illustrated in FIG. 3B. Accordingly, this enables a device with a single page display environment, such as the requesting one of the user client systems 102(1)-102(n), to access and utilize the functionality of a popup window which previously was not possible.

[0048] Referring to FIGS. 3C, 3D, and 6, in step 217 a user at one of the user client systems 102(1)-102(n) enters a value in a search field and activates an associated "Search " button which transmits the request to the web content optimization system 100. In this particular example, the user at one of the user

client systems 102(1)-102(n) enters, "New York, NY" in the search field 132 and hits a "Search " button 134 to transmit the request for the closest airport code or codes to the web content optimization system 100. The web content optimization system 100 transmits the request to the web server system 104 which processes the request. The web server system 104 transmits the results to the web content optimization system 100 which converts the results into one or more fill form links which can be activated to fill a field in the form web page. Next, the web content optimization system 100 inserts the converted results into the popup window 130 on display at the requesting one of the user client systems 102(1)-102(n). In this particular example, in response to the entry of "New York, NY" in the search field 132, the results "JFK/ John F. Kennedy International Airport (JFK) and "LGA/LaGuardia (LGA)" are received, converted to fill form links, and inserted for display in popup window 130 as illustrated in FIG. 3C.

[0049] By way of example only, an HTML fragment corresponding to a fill-form link to fill the "From" field 118 in this example is illustrated below:

JFK/John F Kennedy International (JFK)

[0050] The link contains an un_jtt_fill_param parameter whose value will be used to fill the "From" field 118 in this example. The number of un_jtt_fill_param parameters must correspond to the number of form fields previously indicated in the save-form button. The fill-form link does not contain any data referring to the form page.

[0051] In step 218 a user at one of the user client systems 102(1)-102(n) selects a result link and transmits the request for receipt by the web content optimization system 100. In this particular example, the user at one of the user client systems 102(1)-102(n) selects "JFK/John F Kennedy International (JFK)" to transmit the request which is received by the web content optimization system 100

[0052] In step 220, the web content optimization system 100 determines if the request is selecting a fill form link for filling a value in field in the form web

page. If the web content optimization system 100 determines the request is not selecting a value for filling a field in the airline booking web page 110, then the No branch is taken to step 230. If the web content optimization system 100 determines the request is selecting a value for filling a field in the airline booking web page 110, then the Yes branch is taken to step 222.

[0053] In step 222, the web content optimization system 100 retrieves the previously saved form data. In this particular example, the web content optimization system 100 retrieves the previously entered value for the "date" field 122, the other field values are empty.

[0054] In step 224, the web content optimization system 100 retrieves the selected value for the corresponding entry field. In this particular example, the web content optimization system 100 retrieves "JFK" for the "From" field 118.

[0055] In step 226, the web content optimization system 100 retrieves the saved web form page. In this particular example, the web content optimization system 100 retrieves the airlines booking page 110.

[0056] In step 228, the web content optimization system 110 merges retrieved and modified data of steps 222 and 224 with the previously saved form page of step 226. The requesting one of the user client systems 102(1)-102(n) displays the resulting page in the screen of the display. In this particular example, the web content optimization system 100 inserts "JFK" value into the From field 118 and "10/23/07" into the Date field 122 of the retrieved airlines booking page 110.

[0057] In step 230, the form web page with the stored and selected values is loaded on and shown on the display of the requesting one of the user client systems 102(1)-102(n), although the page can be identified, retrieved, and loaded in other manners. In this particular example, a previously entered date "10/23/07" is entered in the "Date" field 122 and the selected airport code, "JFK" is entered in the "From" field 118 in the airline booking web page 110 which is shown on the display of the requesting one of the user client systems 102(1)-102(n) as illustrated in FIG. 3D.

[0058] Accordingly, as described and illustrated herein the present invention provides web optimization systems and methods which enable the functionality of a popup window inside a web application to be executed in display environments which only permit viewing one page at a time.

[0059] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein.

Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefor, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

What is claimed is:

1. A method for web content optimization, the method comprising:

automatically determining with a web content optimization device in response to an initial request from a client device when a requested web page has an activator for a popup window for at least one of one or more form fields in the requested web page before the requested web page is provided to the client device;

automatically converting with the web content optimization device the activator for the popup window for at least one of one or more form fields in the requested web page to have another functionality from requesting the popup window before the requested web page is provided to the client device;

providing with the web content optimization device the converted web page in response to the initial request;

determining with the web content optimization device when a subsequent request for the popup window as a result of an engagement of the activator in the converted web page is received; and

retrieving and providing with the web content optimization device the requested popup window as another web page in response to the subsequent request.

2. The method as set forth in claim 1 further comprising saving with the web content optimization the converted web page prior to the providing the converted web page in response to the initial request.

3. The method as set forth in claim 1 further comprising saving with the web content optimization device the converted web page with any-values entered in the one or more form fields of the converted web page when the subsequent request for the popup window is received.

4. The method as set forth in claim 3 further comprising:

obtaining with the web content optimization device one or more resulting values in response to another received request with at least one value in at least one of one or more form fields in the another web page for the requested popup window;

converting with the web content optimization device the one or more resulting values into one or more form fill links;

inserting with the web content optimization device the one or more results into one or more form fill links in the corresponding one of one or more form fields in the another web page for the requested popup window; and

providing in response to the another received request with the web content optimization device the another web page for the requested popup window with the inserted one or more form fill links.

5. The method as set forth in claim 4 further comprising:

determining with the web content optimization device when a different received request is selecting one of the form fill links;

retrieving with the web content optimization device the resulting value associated with the selected one of the form fill links and the saved converted web page with any values entered in the one form fields when the different received request is determined to have the selection of one of the form fill links;

merging with the web content optimization device the retrieved resulting value for the selected one of the form fill links with the saved converted web page with any values into a merged web page; and

providing with the web content optimization device the merged web page.

6. A non-transitory computer readable medium having stored thereon instructions for web content optimization comprising machine executable code which when executed by at least one processor, causes the processor to perform steps comprising:

automatically determining in response to an initial request from a client device when a requested web page has an activator for a popup window for at least one of one or more

form fields in the requested web page before the requested web page is provided to the client device;

automatically converting the activator for the popup window for at least one of one or more form fields in the web page to have another functionality from requesting the popup window before the requested web page is provided to the client device;

determining when a subsequent request for the popup window as a result of an engagement of the activator in the converted web page is received; and

retrieving and providing the requested popup window as another web page in response to the subsequent request.

7. The medium as set forth in claim 6 further comprising saving with the web content optimization the converted web page prior to the providing the converted web page in response to the initial request.

8. The medium as set forth in claim 6 further comprising saving the converted web page with any values entered in the one or more form fields of the converted web page when the subsequent request for the popup window is received.

9. The medium as set forth in claim 8 further comprising:

obtaining one or more resulting values in response to another received request with at least one value in at least one of one or more form fields in the another web page for the requested popup window;

converting the one or more resulting values into one or more form fill links;

inserting the one or more results into one or more form fill links in the corresponding one of one or more form fields in the another web page for the requested popup window; and

providing in response to the another received request the another web page for the requested popup window with the inserted one or more form fill links.

10. The medium as set forth in claim 9 further comprising:

determining when a different received request is selecting one of the form fill links;

retrieving the resulting value associated with the selected one of the form fill links and the saved converted web page with any values entered in the one form fields when the different received request is determined to have the selection of one of the form fill links;

merging the retrieved resulting value for the selected one of the form fill links with the saved converted web page with any values into a merged web page; and

providing the merged web page.

11. A web content optimization apparatus comprising:

one or more processors;

a memory coupled to the one or more processors, the one or more processors configured to execute programmed instructions stored in the memory comprising:

automatically determining in response to an initial request from a client device when a requested web page has an activator for a popup window for at least one of one or more form fields in the requested web page before the requested web page is provided to the client device;

automatically converting the activator for the popup window for at least one of one or more form fields in the web page to have another functionality from requesting the popup window before the requested web page is provided to the client device;

determining when a subsequent request for the popup window as a result of an engagement of the activator in the converted web page is received; and

retrieving and providing the requested popup window as another web page in response to the subsequent request.

12. The apparatus as set forth in claim 11 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for further comprising saving with the web content optimization the converted web page prior to the providing the converted web page in response to the initial request.

13. The apparatus as set forth in claim 11 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for further comprising saving the converted web page with any-values entered in the one or more form fields of the converted web page when the subsequent request for the popup window is received.

14. The apparatus as set forth in claim 13 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for further comprising:

obtaining one or more resulting values in response to another received request with at least one value in at least one of one or more form fields in the another web page for the requested popup window;

converting the one or more resulting values into one or more form fill links;

inserting the one or more results into one or more form fill links in the corresponding one of one or more form fields in the another web page for the requested popup window; and

providing in response to the another received request the another web page for the requested popup window with the inserted one or more form fill links.

15. The apparatus as set forth in claim 14 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for further comprising:

determining when a different received request is selecting one of the form fill links;

retrieving the resulting value associated with the selected one of the form fill links and the saved converted web page with any values entered in the one form fields when the different received request is determined to have the selection of one of the form fill links;

merging the retrieved resulting value for the selected one of the form fill links with the saved converted web page with any values into a merged web page; and

providing the merged web page.

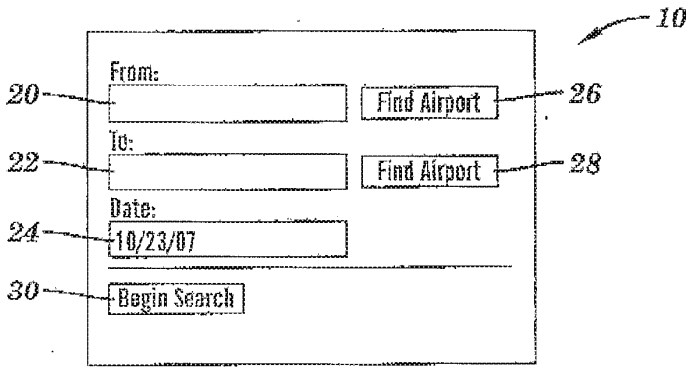


FIG. 1A
PRIOR ART

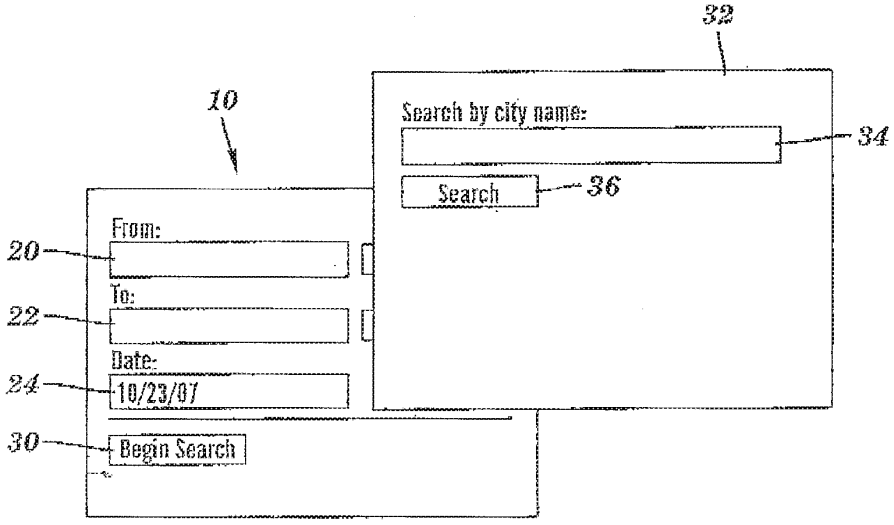


FIG. 1B
PRIOR ART

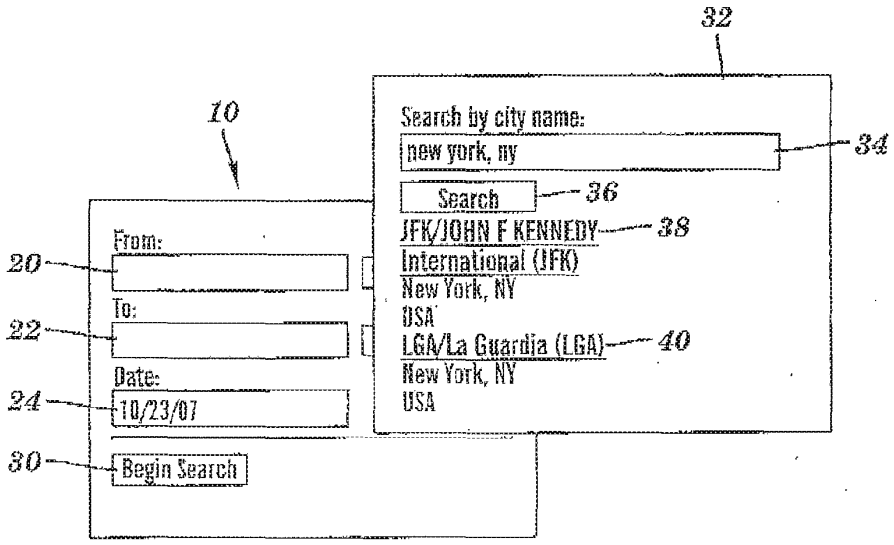


FIG. 1C
PRIOR ART

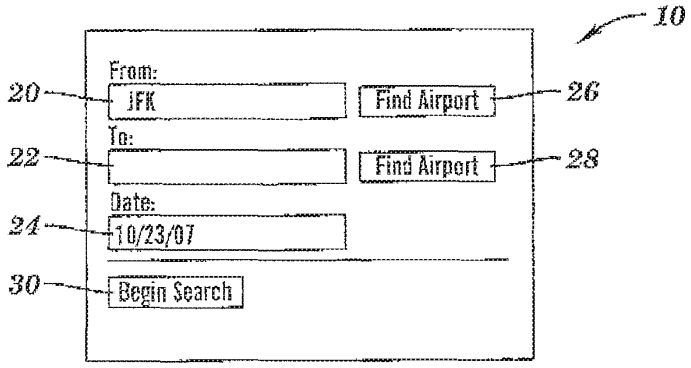


FIG. 1D
PRIOR ART

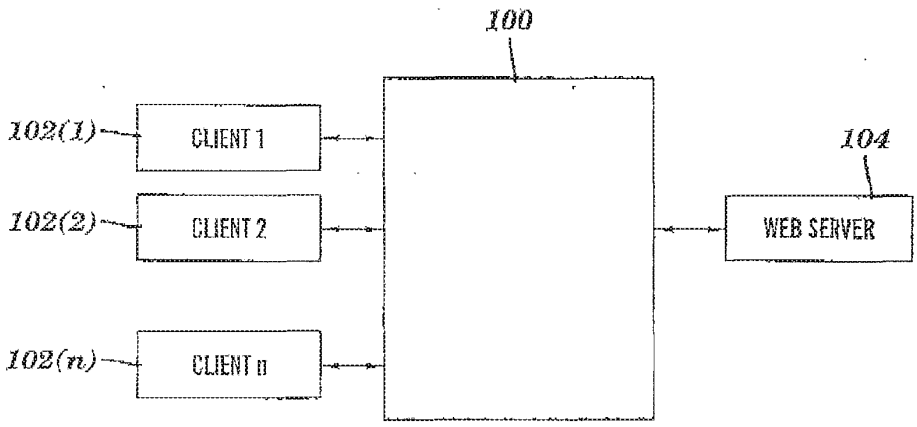


FIG. 2

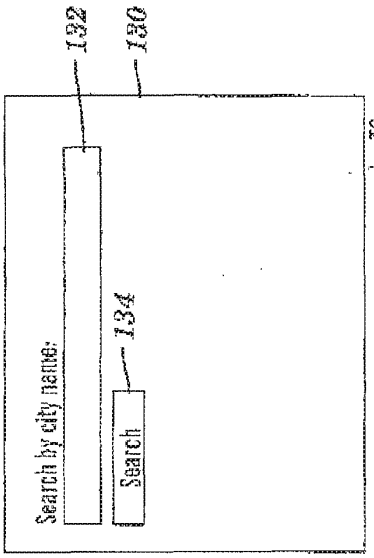


FIG. 3B

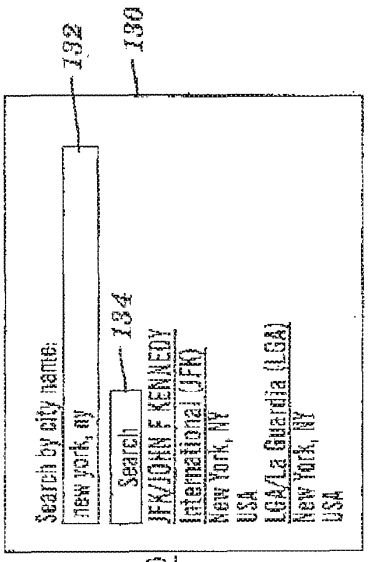


FIG. 3C

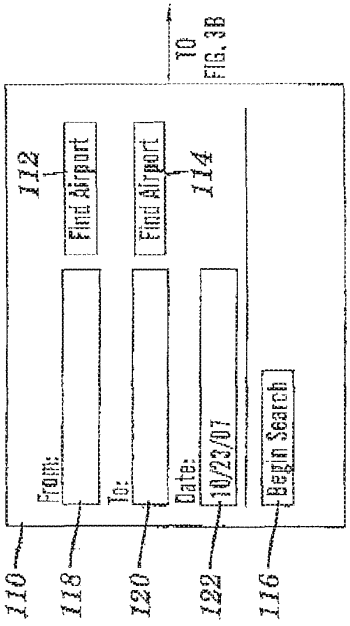


FIG. 3A

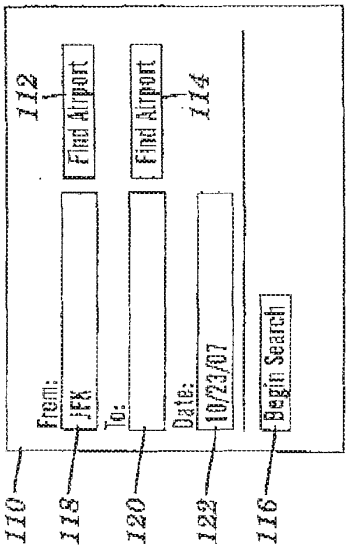


FIG. 3D

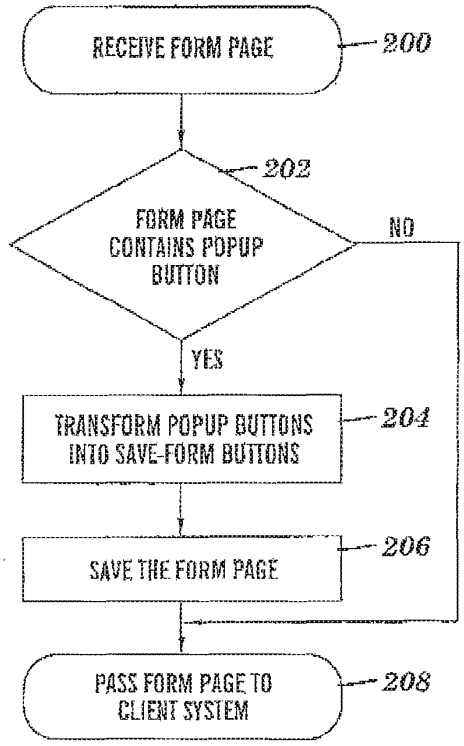

RECEIVE FORM PAGE
200
202
FORM PAGE CONTAINS POPUP BUTTON
NO
YES
TRANSFORM POPUP BUTTONS INTO SAVE-FORM BUTTONS
204
SAVE THE FORM PAGE
206
PASS FORM PAGE TO CLIENT SYSTEM
208

FIG. 4

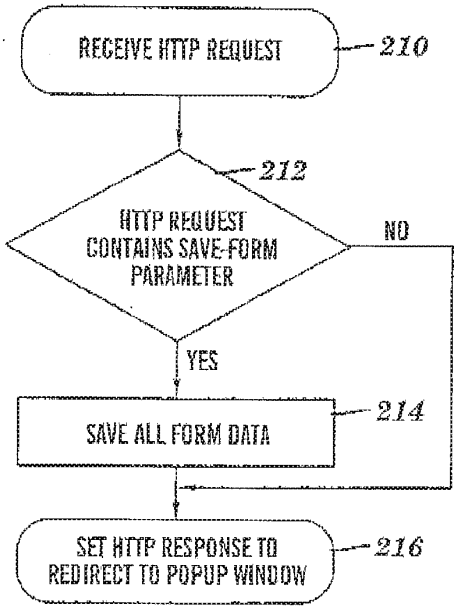

RECEIVE HTTP REQUEST
210
212
HTTP REQUEST CONTAINS SAVE-FORM PARAMETER
NO
YES
SAVE ALL FORM DATA
214
SET HTTP RESPONSE TO REDIRECT TO POPUP WINDOW
216

FIG. 5

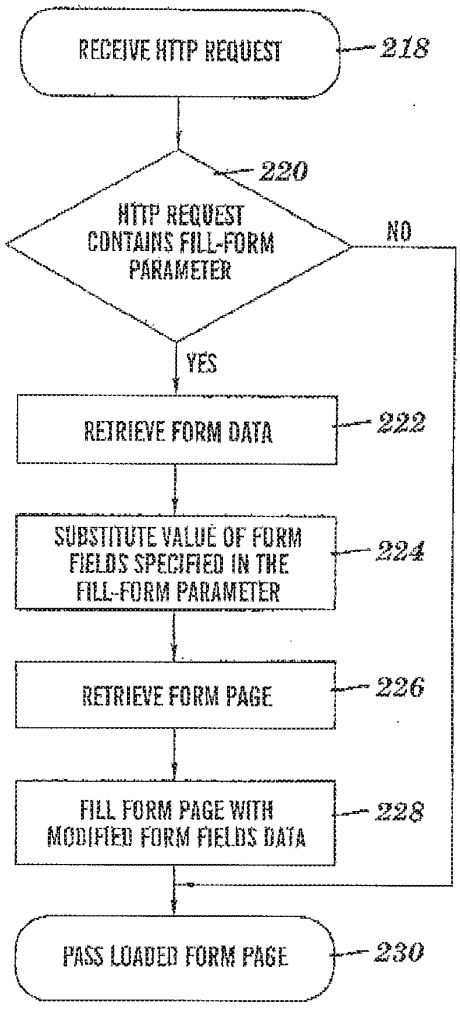
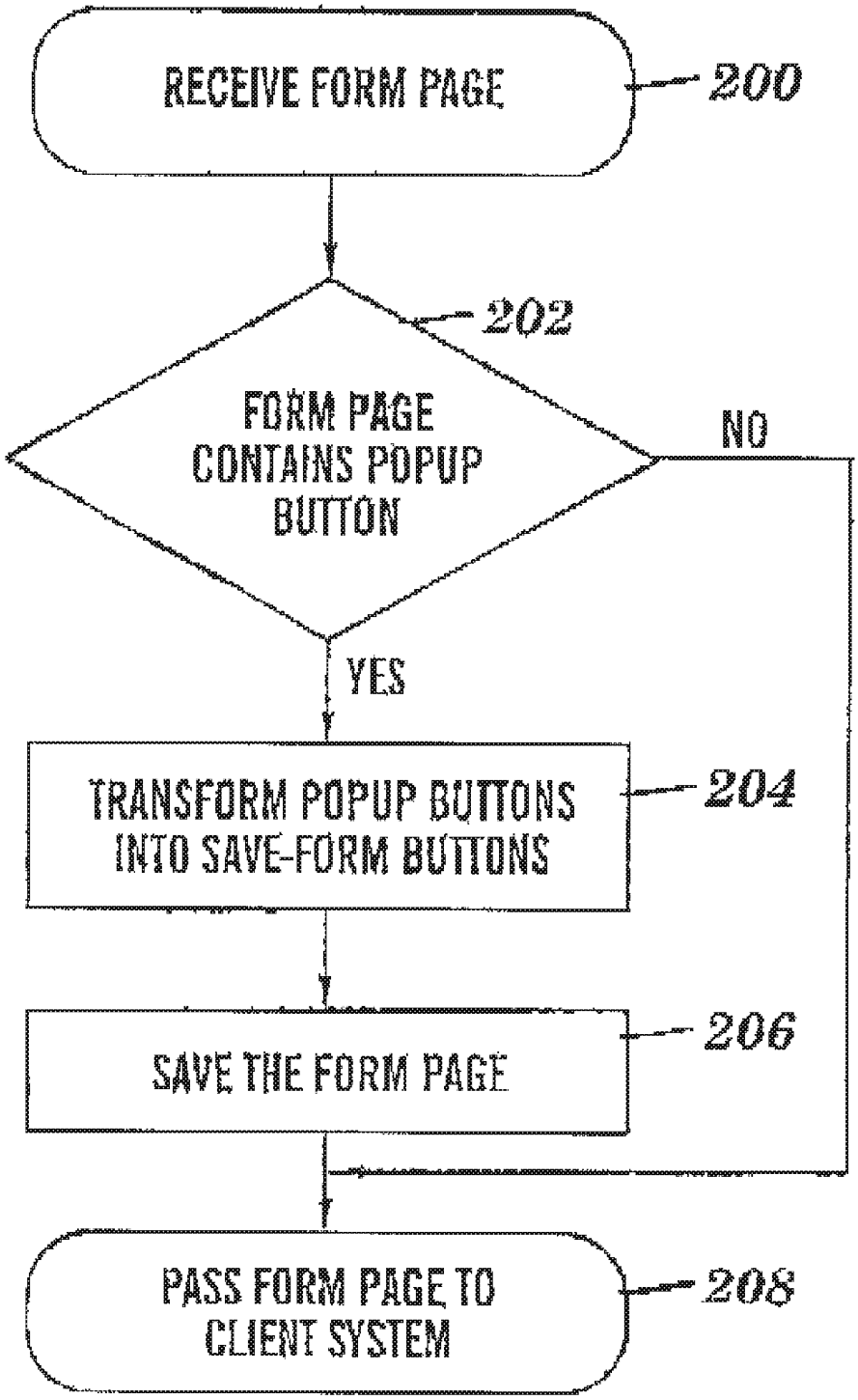

RECEIVE HTTP REQUEST
218
220
HTTP REQUEST CONTAINS FILL-FORM PARAMETER
NO
YES
RETRIEVE FORM DATA
222
SUBSTITUTE VALUE OF FORM FIELDS SPECIFIED IN THE FILL-FORM PARAMETER
224
RETRIEVE FORM PAGE
226
FILL FORM PAGE WITH MODIFIED FORM FIELDS DATA
228
PASS LOADED FORM PAGE
230

FIG. 6


RECEIVE FORM PAGE
200
202
FORM PAGE CONTAINS POPUP BUTTON
NO
YES
TRANSFORM POPUP BUTTONS INTO SAVE-FORM BUTTONS
204
SAVE THE FORM PAGE
206
PASS FORM PAGE TO CLIENT SYSTEM
208

Innovation, Sciences et Développement économique Canada
Office de la Propriété intellectuelle du Canada

Innovation, Science and Economic Development Canada
Canadian Intellectual Property Office

CA 2732540 C 2018/02/27

(11)(21) 2 732 540

(12) BREVET CANADIEN
CANADIAN PATENT

(13) C

(22) Date de dépôt/Filing Date: 2011/02/25

(41) Mise à la disp. pub./Open to Public Insp.: 2011/09/02

(45) Date de délivrance/Issue Date: 2018/02/27

(30) Priorité/Priority: 2010/03/02 (US12/660,637)

(51) Cl.Int./Int.Cl. *H04L 12/16* (2006.01), *H04L 29/06* (2006.01)

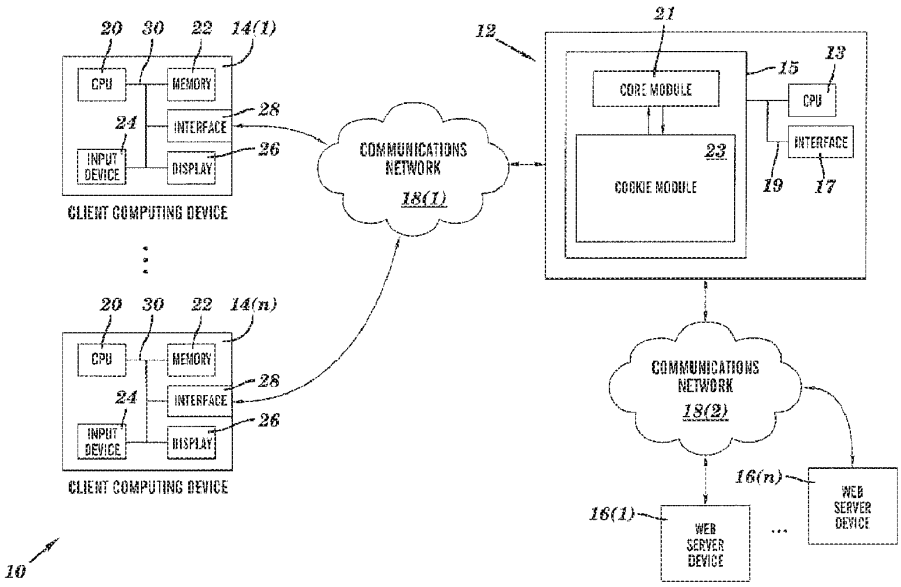
(72) Inventeur/Inventor:
SCODA, ENRICO, IT

(73) Propriétaire/Owner:
USABLENET INC., US

(74) Agent: GOWLING WLG (CANADA) LLP

(54) Titre : PROCEDES D'OPTIMISATION D'UN SERVEUR MANDATAIRE DE CONTENU WEB ET DISPOSITIFS CONNEXES

(54) Title: METHODS FOR OPTIMIZING A WEB CONTENT PROXY SERVER AND DEVICES THEREOF



(57) Abrégé/Abstract:
A method, computer readable medium and apparatus that optimizes a web content proxy server includes obtaining at a proxy server a web page with an original server cookie from one of one or more web content servers in response to a request from one of one or more client devices. A web optimized client cookie based on the original server cookie is generated at the proxy server. The obtained web page with the generated web optimized client cookie is providing by the proxy server to the requesting one of one or more client devices.

ABSTRACT

A method, computer readable medium and apparatus that optimizes a web content proxy server includes obtaining at a proxy server a web page with an original server cookie from one of one or more web content servers in response to a request from one of one or more client devices. A web optimized client cookie based on the original server cookie is generated at the proxy server. The obtained web page with the generated web optimized client cookie is providing by the proxy server to the requesting one of one or more client devices.

METHODS FOR OPTIMIZING A WEB CONTENT PROXY SERVER AND DEVICES THEREOF

FIELD

[0001] This invention generally relates to proxy servers and, more particularly, methods for optimizing web content proxy servers and apparatuses thereof.

BACKGROUND

[0002] A web content optimization server is a proxy server that optimizes web pages obtained from remote web servers for client devices with special requirements, such as mobile phones, PDAs, and smartphones. Every time a client device requests a web page, the web content optimization server downloads the original page from a remote web server, applies some customized rules to extract relevant content, and adapts it to fit the needs of the requesting client device. By way of example, the web content optimization server may remove javascript, linearize content, and adapt the original page to a smaller screen layout for the requesting client device.

[0003] In computing, a cookie, such as a tracking cookie, browser cookie, and HTTP cookie, is a small piece of text stored by a web browser on the client device. A cookie includes one or more name-value pairs containing data, such as user preferences, shopping cart contents, the identifier for a server-based session, or other data used by websites.

[0004] Web content optimization servers need to save cookies to enable the client devices to interact with the original website at the remote web servers in the correct way. Accordingly, web content optimization servers store these cookies in an internal memory and associate them with the corresponding session from each client device so that when the same client device sends a request for a new page, the web content optimization server will load the matching cookies and send them to the remote web server to get the page to process. Unfortunately,

storing the cookies for these client devices causes problems with scalability, security, and privacy of the web content optimization servers.

SUMMARY

[0005] A method for optimizing a web content proxy server includes obtaining at a proxy server a web page with an original server cookie from one of one or more web content servers in response to a request from one of one or more client devices. A web optimized client cookie based on the original server cookie is generated at the proxy server. The obtained web page with the generated web optimized client cookie is providing by the proxy server to the requesting one of one or more client devices.

[0006] A computer readable medium having stored thereon instructions for optimizing a proxy server comprising machine executable code which when executed by at least one processor, causes the processor to perform steps including obtaining a web page with an original server cookie from one or more web servers in response to a request from one of one or more client devices. A web optimized client cookie is generated based on the original server cookie and the obtained web page with the generated web optimized client cookie is transmitted to the requesting one of one or more client devices.

[0007] A web content proxy server or apparatus includes one or more processors and a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory including obtaining a web page with an original server cookie from one or more web servers in response to a request from one of one or more client devices. A web optimized client cookie is generated based on the original server cookie and the obtained web page with the generated web optimized client cookie is transmitted to the requesting one of one or more client devices

[0008] This technology provides a number of advantages including providing a method, computer readable medium and an apparatus that optimizes implementation of a web content proxy server for interactions involving cookies between client devices and remote web servers. With this technology, original

server cookies are transformed by the web content proxy server to web optimized client cookies which are transmitted to the client devices requesting the web pages for storage and use with subsequent requests.

[0009] This technology provides greater scalability because the web optimized client cookies are stored in the web browser at the client device, not in memory at the web content proxy server. As a result, the web content proxy server does not face any issues with respect to memory storage capacity due to the number of sessions with cookies for client devices. The web content proxy server can use the same memory whether there are 100 or 1,000,000 or more client devices engaged in sessions with the remote web servers through the web content proxy server.

[00010] Additionally, this technology provides greater security and privacy because the web content proxy server does not contain a centralized database of original server cookies which contain session information from client devices browsing pages of web sites. Instead, these original server cookies are translated into web optimized client cookies which are then dispersed out among the client devices. As a result, the web content proxy server does not have any stored cookies from interactions between client devices and remote web servers that could be used to steal identity or other confidential information of these client devices.

In accordance with an aspect of the present disclosure there is provided a method implemented by one or more web content proxy servers, the method comprising: obtaining content with an original server cookie from one of one or more web content servers in response to a request from one of one or more client devices; generating a web optimized client cookie comprising a name comprising at least a name of the original server cookie and at least a portion of a domain and a path associated with the obtained content; providing the obtained content with the generated web optimized client cookie to the requesting one of one or more client devices; and reconstituting the original server cookie using the web optimized client cookie included in a subsequent request received from the one of the one or more client devices.

In accordance with an aspect of the present disclosure there is provided a non-transitory computer readable medium having stored thereon instructions for optimizing a proxy server comprising machine executable code which when executed by at least one processor, causes the processor to: obtain content with an original server cookie from one of one or more web content servers in response to a request from one of one or more client devices; generate a web optimized client cookie comprising a name comprising at least a name of the original server cookie and at least a portion of a domain and a path associated with the obtained content; provide the obtained content with the generated web optimized client cookie to the requesting one of one or more client devices; and reconstitute the original server cookie using the web optimized client cookie included in a subsequent request received from the one of the one or more client devices.

In accordance with an aspect of the present disclosure there is provided a web proxy apparatus comprising: one or more processors; a memory comprising programmed instructions stored thereon, the memory coupled to the one or more processors which are configured to execute the store programmed instructions to: obtain content with an original server cookie from one of one or more web content servers in response to a request from one of one or more client devices; generate a web optimized client cookie comprising a name comprising at least a name of the original server cookie and at least a portion of a domain and a path associated with the obtained content; provide the obtained content with the generated web optimized client cookie to the requesting one of one or more client devices; and reconstitute the original server cookie using the web optimized client cookie included in a subsequent request received from the one of the one or more client devices.

BRIEF DESCRIPTION OF THE DRAWINGS

[00011] FIG. 1 is a block diagram of an exemplary environment with an optimized web content proxy server;

[00012] FIG. 2A is an example of a HTTP request for a web page from a remote web server;

[00013] FIG. 2B is an example of a HT-FP response with an original server cookie from a remote web server to a HTTP request;

[00014] FIG. 2C is an example of a HTTP response containing the web optimized client cookie generated from the original server cookie received shown in FIG. 2B;

[00015] FIG. 2D is an example of another HTTP request with the web optimized client cookie shown in FIG. 2C for a web page from a remote web server.

[00016] FIG. 2E is an example of the another HTTP request with the web optimized client cookie shown in FIG. 2D translated into the original server cookie for transmission to the remote web server with the another get request;

[00017] FIG. 3 is a flow chart of an example of a method for generating a web optimized client cookie from an original server cookie to optimize implementation of a web content proxy server; and

[00018] FIG. 4 is a flow chart of an example of a method for transforming a web optimized client cookie back to an original server cookie to optimize implementation of a web content proxy server.

DETAILED DESCRIPTION

[00019] An exemplary environment 10 in which a web content proxy server 12 is optimized is illustrated in FIG. 1. The exemplary environment 10 includes a web content proxy server or apparatus 12, client devices 14(1)-14(n), web server devices 16(1)-16(n), and communication networks 18(1)-18(2), although other numbers and types of systems, devices, and/or elements in other configurations and environments with other communication network topologies can be used. This technology provides a number of advantages including providing a method, computer readable medium and an apparatus that optimizes implementation of a web content proxy server for interactions involving cookies between client devices and remote web servers.

[00020] Referring more specifically to FIG. 1, the web content proxy server 12 optimizes the handling of original server cookies from the web server devices 16(1)-16(n) for requesting client devices 14(1)-14(n) and the handling of

web optimized client cookies, although the web content proxy server 12 can provide other numbers and types of functions. Although one web content proxy server 12 is shown, other numbers and types of web content proxy devices and systems can be used.

[00021] The web content proxy server 12 includes a central processing unit (CPU) or processor 13, a memory 15, and an interface system 17 which are coupled together by a bus 19 or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor 13 in the web content proxy server 12 executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[00022] The memory 15 in the web content proxy server 12 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor 13, can be used for the memory 15 in the web content proxy server 12. In these embodiments, the memory 15 includes a core module 21 and a cookie module 23 which store programmed instructions for one or more aspects of the present invention as described and illustrated herein, although the memory can comprise other types and numbers of systems, devices, and elements in other configurations which store other data. The cookie module 23 includes programmed instructions and/or logic configured to translate an original server cookie into a web optimized client cookie and to extract the original server cookie when a web optimized client cookie is received, although the cookie module 23 can have other types and numbers of functions as described and illustrated herein.

[00023] The interface system 17 in the web content proxy server 12 is used to operatively couple and communicate between the web content proxy server 12 and the client devices 14(1)-14(n) and the web server devices 16(1)-16(n) via the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used. By way of example only, the communication networks 18(1) and 18(2) can use TCP/IP over Ethernet and industry-standard protocols, including HTTP, HTTPS, WAP, and SOAP, although other types and numbers of communication networks, such as a direct connection, a local area network, a wide area network, modems and phone lines, e-mail, and wireless and hardwire communication technology, each having their own communications protocols, can be used.

[00024] Each of the client devices 14(1)-14(n) enables a user to request, get and interact with web pages from one or more web sites hosted by the web server devices 16(1)-16(n) through the web content proxy server 12 via one or more communication networks, although one or more of the client devices 14(1)-14(n) could access content and utilize other types and numbers of applications from other sources and could provide a wide variety of other functions for the user. Although multiple client devices 14(1)-14(n) are shown, other numbers and types of user computing systems could be used. In this example, the client devices 14(1)-14(n) comprise mobile devices with Internet access that permit a website form page or other retrieved data to be displayed, although each of the client devices 14(1)-14(n). By way of example only, one or more of the client devices 14(1)-14(n) can comprise smart phones, personal digital assistants, or computers.

[00025] Each of client devices 14(1)-14(n) in this example is a computing device that includes a central processing unit (CPU) or processor 20, a memory 22, user input device 24, a display 26, and an interface system 28, and which are coupled together by a bus 30 or other link, although one or more of client devices 14(1)-14(n) can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 20 in each of client devices 14(1)-14(n) executes a program of stored instructions for one or more

aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions.

[00026] The memory 22 in each of the client devices 14(1)-14(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein as well as the web optimized client cookies, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor 20 can be used for the memory 22 in each of the client devices 14(1)-14(n.

[00027] The user input device 24 in each of the client devices 14(1)-14(n) is used to input selections, such as requests for a particular website form page or to enter data in fields of a form page, although the user input device could be used to input other types of data and interact with other elements. The user input device can include keypads, touch screens, and/or vocal input processing systems although other types and numbers of user input devices can be used.

[00028] The display 26 in each of the client devices 14(1)-14(n) is used to show data and information to the user, such as website page by way of example only. The display in each of the client devices 14(1)-14(n) is a phone screen display, although other types and numbers of displays could be used depending on the particular type of client device.

[00029] The interface system 28 in each of the client devices 14(1)-14(n) is used to operatively couple and communicate between the client devices 14(1)-14(n) and the web content proxy server 12 and web server devices 16(1)-16(n) over the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00030] The web server devices 16(1)-16(n) provide one or more pages from one or more web sites for use by one or more of the client devices 14(1)-14(n) via the web content proxy server 12, although the web server devices 16(1)-16(n) can provide other numbers and types of applications and/or content and can have provide other numbers and types of functions. Although web server devices 16(1)-16(n) are shown for ease of illustration and discussion, other numbers and types of web server systems and devices can be used.

[00031] Each of the web server devices 16(1)-16(n) include a central processing unit (CPU) or processor, a memory, and an interface system which are coupled together by a bus or other link, although each of the web server devices 16(1)-16(n) could have other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor in each of the web server devices 16(1)-16(n) executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[00032] The memory in each of the web server devices 16(1)-16(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated by way of the embodiments, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor, can be used for the memory in each of the web server devices 16(1)-16(n).

[00033] The interface system in each of the web server devices 16(1)-16(n) is used to operatively couple and communicate between the web server devices 16(1)-16(n) and the web content proxy server 12 and the client devices 14(1)-14(n) via communication networks 18(1) and 18(2), although other types and

numbers of communication networks with other types and numbers of connections and configurations can be used.

[00034] Although embodiments of the web content proxy server 12, the client devices 14(1)-14(n), and the web server devices 16(1)-16(n), are described and illustrated herein, each of the client devices 14(1)-14(n), the web content proxy server 12, and the web server devices 16(1)-16(n), can be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s).

[00035] Furthermore, each of the systems of the embodiments may be conveniently implemented using one or more general purpose computer systems, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[00036] In addition, two or more computing systems or devices can be substituted for any one of the systems in any embodiment of the embodiments. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the embodiments. The embodiments may also be implemented on computer system or systems that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[00037] The embodiments may also be embodied as a computer readable medium having instructions stored thereon for one or more aspects of the present invention as described and illustrated by way of the embodiments herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the embodiments, as described and illustrated herein.

[00038] An exemplary method for generating a web optimized client cookie from an original server cookie to optimize implementation of the web content proxy server 12 will now be described with reference to FIGS. 1-2C and 3. In step 50, in this example one of the client devices 14(1)-14(n) via a web browser requests a page A.html at the website, "www.example.com" as shown in one example in FIG. 2A. This request is transmitted to the web content proxy server 12 which processes and transmits the request to the one of the web servers 16(1)-16(n) hosting the website "www.example.com." The hosting one of the web servers 16(1)-16(n) provides a response in this example for the requested page A.html which also contains an original server cookie "SESSION" to the web content proxy server 12 as shown in FIG. 2B. In this example, SESSION has a value equal to "1234", the domain is equal to ".example.com" and the path is equal to "/". This response uses the HTTP header Field "Set-Cookie". The cookie is a string formed by the pair "name=value", followed by optional attributes, like those in this example indicating the server domain(s) and path accepting this cookie. Although one illustrative example is described herein, this technology can be used with specifications for all cookies.

[00039] Next, in step 52 the web content proxy server 12 determines whether the original server cookie includes the domain attribute for the requested web page provided by the hosting one of the web servers 16(1)-16(n). If in step 52 the web content proxy server 12 determines the original server cookie does not include the domain attribute, then the No branch is taken to step 54. In step 54, the web content proxy server 12 extracts the domain attribute from the requested web page provided by the hosting one of the web servers 16(1)-16(n). If in step 52 the web content proxy server 12 determines the original server cookie does include the domain attribute, then the Yes branch is taken to step 56.

[00040] In step 56, the web content proxy server 12 determines whether the original server cookie includes the path attribute for the requested web page provided by the hosting one of the web servers 16(1)-16(n). If in step 52 the web content proxy server 12 determines the original server cookie does not include the path attribute, then the No branch is taken to step 58. In step 58, the web content proxy server 12 extracts the path attribute from the requested web page provided by the hosting one of the web servers 16(1)-16(n). If in step 56 the web content proxy server 12 determines the original server cookie does include the path attribute, then the Yes branch is taken to step 60.

[00041] In step 60, the web content proxy server 12 generates a name for the new web optimized client cookie by concatenating the original name with domain and path, separated by spaces, although other manners for generating the new name can be used. The resulting new name is "universal resource locator encoded" to keep conformance to the cookie specification. Additionally, the resulting new name is unique even if different domains contain cookies with the same name. This new name contains all the information necessary for the web content proxy server 12 to extract the original server cookie later as described in greater detail by reference to FIG. 4,

[00042] In step 62, the web content proxy server 12 forms the new web optimized client cookie by associating the new name with the same value of the original server cookie, with the domain attribute not being specified, and with the path attribute being associated with a value "/", although other values can be used, such as one for the path attribute that corresponds to a prefix associated with this optimization method (by way of example only "/mt/").

[00043] By way of example only, when the web content proxy server 12 receives a response with the original server cookie as shown in FIG. 2B, the web content proxy server 12 generates a web optimized client cookie as shown in FIG. 2C. More specifically, the original server cookie: SESSION=1234; domain=.example.com; and path=/ is transformed by the web content proxy server 12 to a web optimized client cookie: SESSION+.example.com+%252F=1234; path=/mt/. Accoridingly, in this illustrative example the new web optimized client

cookie name SESSION+.example.com+%252F is the encoded version of the concatenation of original server cookie name, domain attribute and path attribute, although other orders and manners for forming this name can be used. In this example, the new path attribute corresponds to a prefix "/mt/" associated with this optimization method.

[00044] Next, in step 64 the web content proxy server 12 copies the remaining attributes in the original server cookie, such as an expiration date for the original server cookie by way of example, in the web optimized client cookie, although other amounts of the remaining attributes could be copied and other information also could be added.

[00045] Next, in step 66 the original server cookie which has been translated into the web optimized client cookie is now provided to the core module 21 in the web content proxy server 12. The core module 21 includes programmed instructions and/or logic to manage the transmission of the web optimized client cookie from the web content proxy server 12 to the requesting one of the client devices 14(1)-14(n). The web browser at the requesting one of the client devices 14(1)-14(n) receives and saves the web optimized client cookie in the memory 22 at the requesting one of the client devices 14(1)-14(n). In this illustrative example, the web optimized client cookie shown in FIG. 2C is stored in the memory 22 at the requesting one of the client devices 14(1)-14(n).

[00046] Referring now to FIGS. 1, 2D-2E and 4, an exemplary method for translating a web optimized client cookie back to an original server cookie to optimize the implementation of the web content proxy server 12 will now be described. In step 100, in this example one of the client devices 14(1)-14(n) via a web browser submits another request to the web content proxy server 12 for page B.html at the website, "www.example.com" as shown in one example in FIG. 2D. This request includes a web optimized client cookie which in this example comprises SESSION+.example.com+%252F.

[00047] In step 102, the web content proxy server 12 extracts the original server cookie name and the domain and path attributes from the name of the web

optimized client cookie. In this illustrative example, the original server cookie name and the domain and path attributes are extracted by the web content proxy server from the name: SESSION+.example.com+%252F.

[00048] In step 104, the web content proxy server 12 determines whether the extracted domain and path attributes identify a web optimized client cookie that is a match to universal resource locator for the requested web page. If in step 104 the web content proxy server 12 determines the extracted domain and path attributes identify a web optimized client cookie is not a match, then the No branch is taken to step 106. In step 106, the web content proxy server 12 submits the request to the hosting one of the web servers 16(1)-16(n) hosting the request page without an original server cookie. In this illustrative example, the requested page is "B.html." If in step 104 the web content proxy server 12 determines the extracted domain and path attributes identify a web optimized client cookie is a match, then the Yes branch is taken to step 108.

[00049] In step 108, the web content proxy server 12 associates the extracted name from the web optimized client cookie with the value for the original server cookie. The extracted name and value comprise the original server cookie which is appended to the HTTP cookie header fields of the request to be sent to the one of the web servers 16(1)-16(n) hosting the requested web page. In this illustrative example, the extracted name SESSION is associated with the value 1234.

[00050] In step 110, the web content proxy server 12 submits the request with the reconstituted original server cookie to the one of the web servers 16(1)-16(n) hosting the requested page. In this illustrative example, the request with the reconstituted original server cookie as shown in FIG. 2E is transmitted to the one of the web servers 16(1)-16(n) hosting the requested page.

[00051] Accordingly, as illustrated and described herein this technology provides a number of advantages including providing a method, computer readable medium and an apparatus that optimizes implementation of a web content proxy server for interactions involving cookies between client devices and

remote web servers. With this technology, the web content proxy server is much more scalable because of the reduced memory storage demands. Additionally, with this technology the web content proxy server poses a much lower security and privacy risk to information provided by the client devices 14(1)-14(n).

[00052] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

What is claimed:

1. A method implemented by one or more web content proxy servers, the method comprising:

obtaining content with an original server cookie from one of one or more web content servers in response to a request from one of one or more client devices;

generating a web optimized client cookie comprising a name comprising at least a name of the original server cookie and at least a portion of a domain and a path associated with the obtained content;

providing the obtained content with the generated web optimized client cookie to the requesting one of one or more client devices; and

reconstituting the original server cookie using the web optimized client cookie included in a subsequent request received from the one of the one or more client devices.

2. The method as set forth in claim 1 further comprising providing the obtained content and the web optimized client cookie to the requesting one of the one or more client devices without storing the original server cookie or the web optimized client cookie in a local memory.

3. The method as set forth in claim 1 further comprising obtaining the domain and path attributes of the obtained content from a uniform resource locator (URL) for the obtained content.

4. The method as set forth in claim 1 further comprising:

concatenating the name of the original server cookie and the at least a portion of the domain and the path associated with the obtained content to form the name of the web optimized client cookie; and

associating the name of the web optimized client cookie with a value of the original server cookie.

5. The method as set forth in claim 1 further comprising appending one or more original attributes of the original server cookie to the web optimized client cookie.

6. The method as set forth in claim 1 further comprising:

determining when the reconstituted original server cookie corresponds with a network address of the subsequent request; and

providing the subsequent request with the reconstituted original server cookie to the one of the one or more web content servers when the determining indicates that the reconstituted original server cookie corresponds with the network address of the subsequent request.

7. The method as set forth in claim 6 further comprising associating one or more original attributes of the original server cookie to the reconstituted original server cookie.

8. A non-transitory computer readable medium having stored thereon instructions for optimizing a proxy server comprising machine executable code which when executed by at least one processor, causes the processor to:

obtain content with an original server cookie from one of one or more web content servers in response to a request from one of one or more client devices;

generate a web optimized client cookie comprising a name comprising at least a name of the original server cookie and at least a portion of a domain and a path associated with the obtained content;

provide the obtained content with the generated web optimized client cookie to the requesting one of one or more client devices; and

reconstitute the original server cookie using the web optimized client cookie included in a subsequent request received from the one of the one or more client devices.

9. The non-transitory computer readable medium as set forth in claim 8 wherein the machine executable code when executed by the processor further cause the processor to provide the obtained content and the web optimized client cookie to the requesting one of the one or more client devices without storing the original server cookie or the web optimized client cookie in a local memory.

10. The non-transitory computer readable medium as set forth in claim 9 wherein the machine executable code when executed by the processor further cause the processor to obtain the domain and path attributes of the obtained content from a uniform resource locator (URL) for the obtained content.

11. The non-transitory computer readable medium as set forth in claim 9 wherein the machine executable code when executed by the processor further cause the processor to:

concatenate the name of the original server cookie and the at least a portion of the domain and the path associated with the obtained content to form the name of the web optimized client cookie; and

associate the name of the web optimized client cookie with a value of the original server cookie.

12. The non-transitory computer readable medium as set forth in claim 9 wherein the machine executable code when executed by the processor further cause the processor to append one or more original attributes of the original server cookie to the web optimized client cookie.

13. The non-transitory computer readable medium as set forth in claim 8 wherein the machine executable code when executed by the processor further cause the processor to:

determine when the reconstituted original server cookie corresponds with a network address of the subsequent request; and

provide the subsequent request with the reconstituted original server cookie to the one of the one or more web content servers when the determining indicates that the reconstituted original server cookie corresponds with the network address of the subsequent request.

14. The non-transitory computer readable medium as set forth in claim 13 wherein the machine executable code when executed by the processor further cause the processor to associate one or more original attributes of the original server cookie to the reconstituted original server cookie.

15. A web proxy apparatus comprising:

one or more processors;

a memory comprising programmed instructions stored thereon, the memory coupled to the one or more processors which are configured to execute the store programmed instructions to:

obtain content with an original server cookie from one of one or more web content servers in response to a request from one of one or more client devices;

generate a web optimized client cookie comprising a name comprising at least a name of the original server cookie and at least a portion of a domain and a path associated with the obtained content;

provide the obtained content with the generated web optimized client cookie to the requesting one of one or more client devices; and

reconstitute the original server cookie using the web optimized client cookie included in a subsequent request received from the one of the one or more client devices.

16. The apparatus as set forth in claim 15 wherein the one or more processors are further configured to execute the stored programmed instructions to provide the obtained content and the web optimized client cookie to the requesting one of the one or more client devices without storing the original server cookie or the web optimized client cookie in the memory.

17. The apparatus as set forth in claim 16 wherein the one or more processors are further configured to execute the stored programmed instructions to obtain the domain and path attributes of the obtained content from a uniform resource locator (URL) for the obtained content.

18. The apparatus as set forth in claim 16 wherein the one or more processors are further configured to execute the stored programmed instructions to;

concatenate the name of the original server cookie and the at least a portion of the domain and the path associated with the obtained content to form the name of the web optimized client cookie; and

associate the name of the web optimized client cookie with a value of the original server cookie.

19. The apparatus as set forth in claim 16 wherein the one or more processors are further configured to execute the stored programmed instructions to append one or more original attributes of the original server cookie to the web optimized client cookie.

20. The apparatus as set forth in claim 15 wherein the one or more processors are further configured to execute the stored programmed instructions to:

determine when the reconstituted original server cookie corresponds with a network address of the subsequent request; and

provide the subsequent request with the reconstituted original server cookie to the one of the one or more web content servers when the determining indicates that the reconstituted original server cookie corresponds with the network address of the subsequent request.

21. The apparatus as set forth in claim 20 wherein the one or more processors are further configured to execute the stored programmed instructions to associate one or more original attributes of the original server cookie to the reconstituted original server cookie.

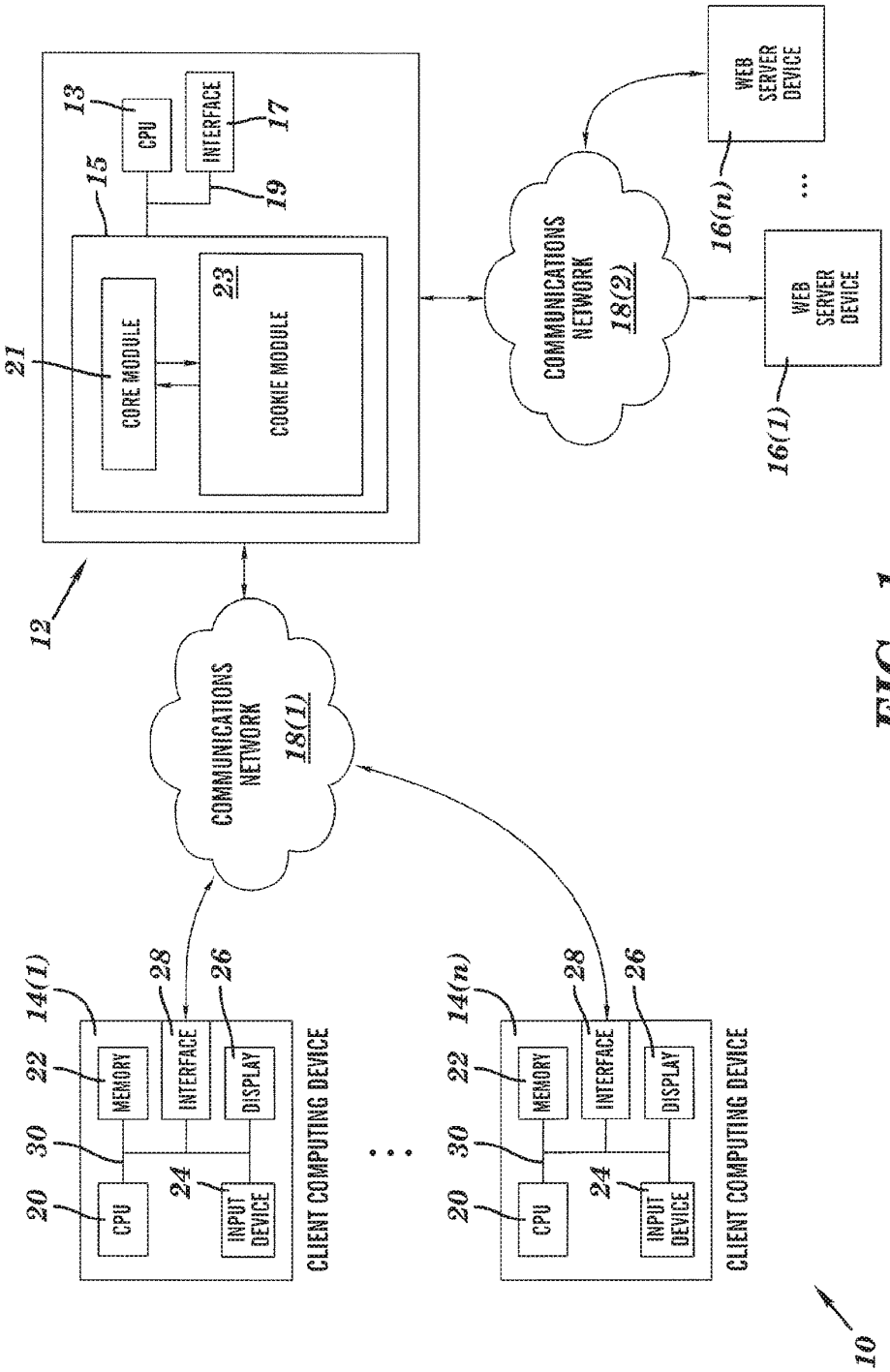

10
12
13
CPU
INTERFACE
17
15
19
21
CORE MODULE
23
COOKIE MODULE
COMMUNICATIONS NETWORK 18(1)
COMMUNICATIONS NETWORK 18(2)
WEB SERVER DEVICE
16(n)
16(1)
14(1)
14(n)
22
MEMORY
28
INTERFACE
26
DISPLAY
30
20
CPU
24
INPUT DEVICE
CLIENT COMPUTING DEVICE

FIG. 1

```
GET /A.html HTTP/1.1
Host: www.example.com
Accept: */*
User-Agent: my-mobile-browser 1.0
```

FIG. 2A

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Content-Length: 5300
Set-Cookie: SESSION=1234; domain=.example.com; path=/
```

FIG. 2B

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Content-Length: 5300
Set-Cookie: SESSION+.example.com+%252F=1234; path=/mt/
```

FIG. 2C

```
GET /mt/www.example.com/B.html HTTP/1.1
User-Agent: my-mobile-browser 1.0
Host: m.proxy.com
Accept: */*
Cookie: SESSION+.example.com+%252F=1234
```

FIG. 2D

```
GET /B.html HTTP/1.1
User-Agent: my-mobile-browser 1.0
Host: www.example.com
Accept: */*
Cookie: SESSION=1234
```

FIG. 2E

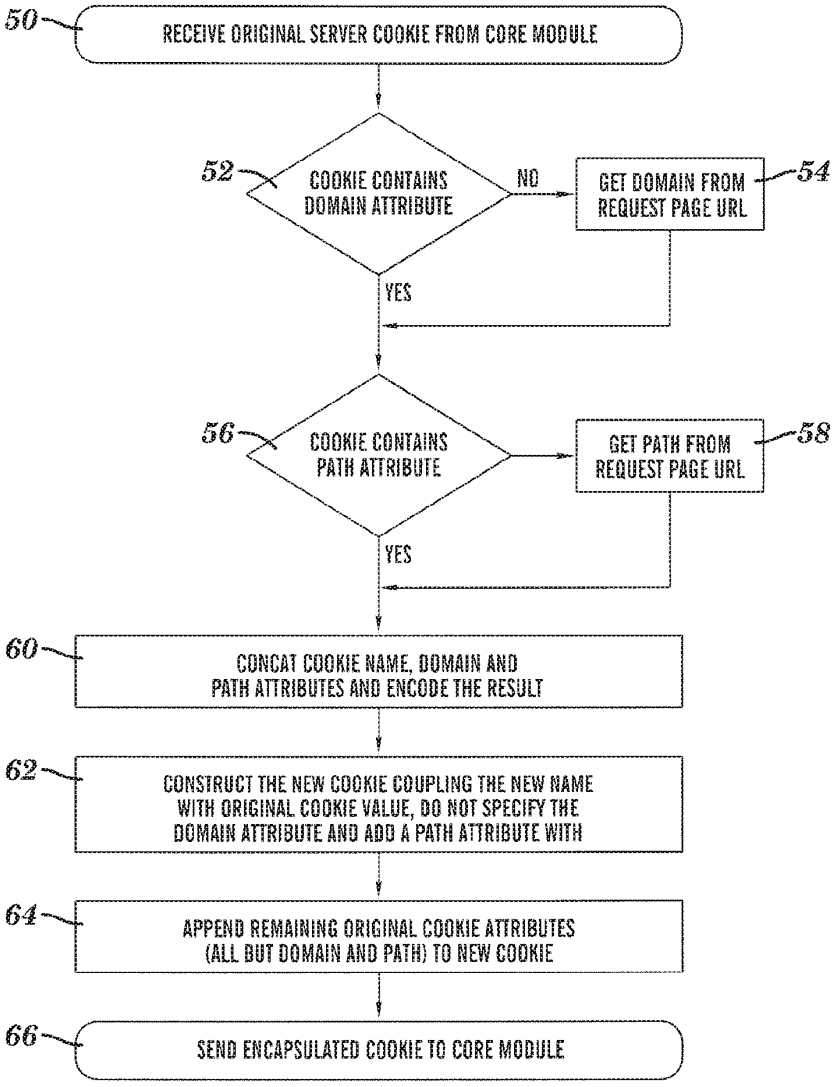

50
RECEIVE ORIGINAL SERVER COOKIE FROM CORE MODULE
52
COOKIE CONTAINS DOMAIN ATTRIBUTE
NO
GET DOMAIN FROM REQUEST PAGE URL
54
YES
56
COOKIE CONTAINS PATH ATTRIBUTE
GET PATH FROM REQUEST PAGE URL
58
YES
60
CONCAT COOKIE NAME, DOMAIN AND PATH ATTRIBUTES AND ENCODE THE RESULT
62
CONSTRUCT THE NEW COOKIE COUPLING THE NEW NAME WITH ORIGINAL COOKIE VALUE, DO NOT SPECIFY THE DOMAIN ATTRIBUTE AND ADD A PATH ATTRIBUTE WITH
64
APPEND REMAINING ORIGINAL COOKIE ATTRIBUTES (ALL BUT DOMAIN AND PATH) TO NEW COOKIE
66
SEND ENCAPSULATED COOKIE TO CORE MODULE

FIG. 3

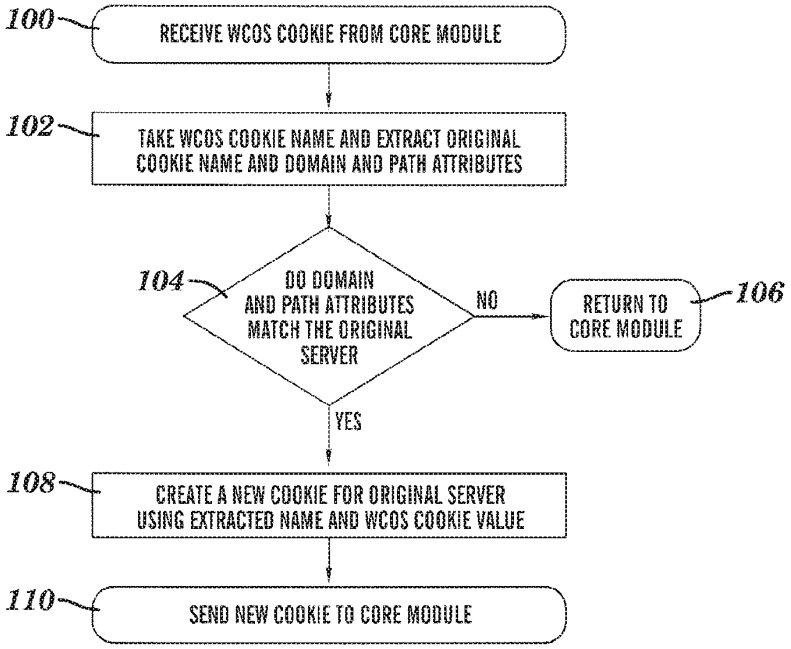
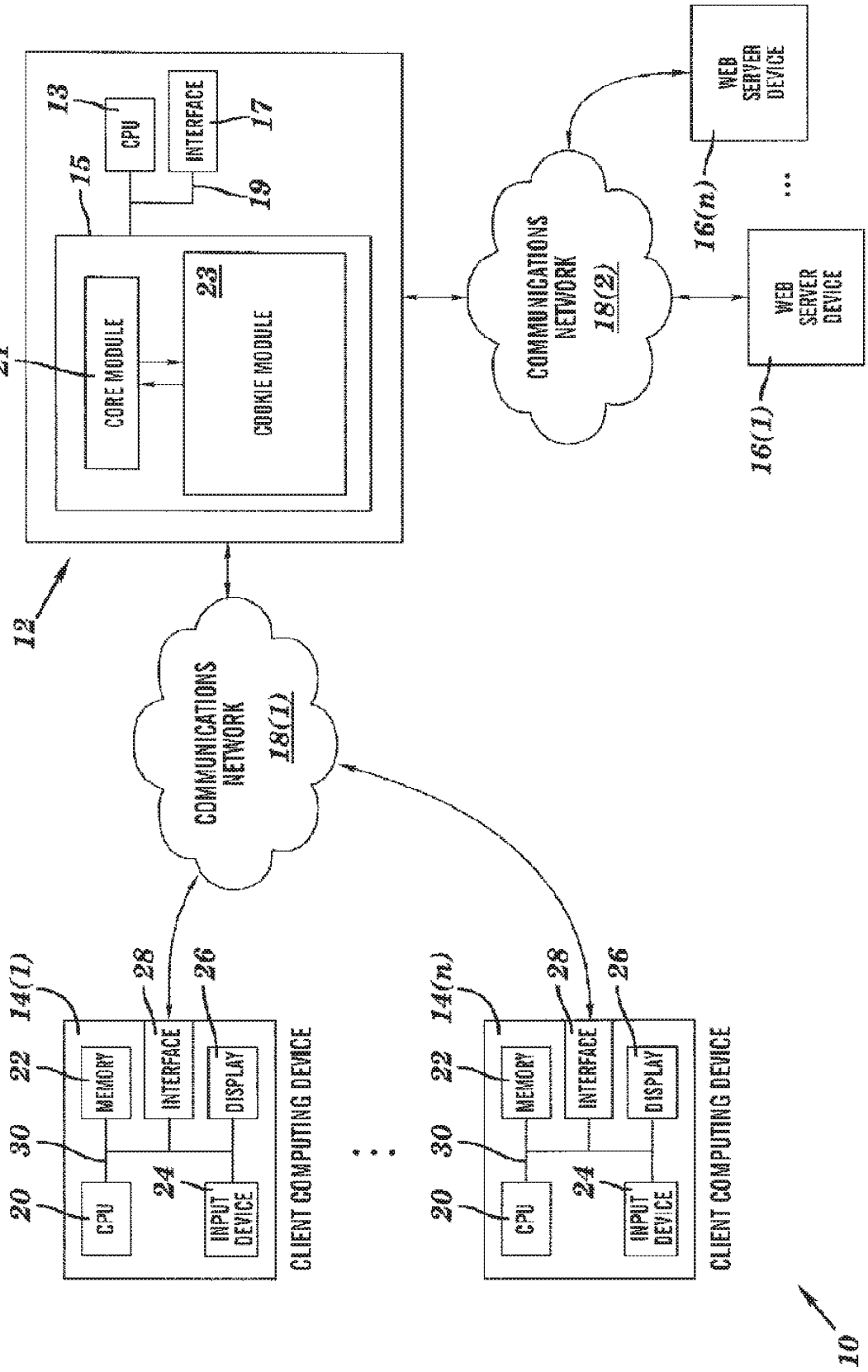

100
RECEIVE WCOS COOKIE FROM CORE MODULE
102
TAKE WCOS COOKIE NAME AND EXTRACT ORIGINAL COOKIE NAME AND DOMAIN AND PATH ATTRIBUTES
104
DO DOMAIN AND PATH ATTRIBUTES MATCH THE ORIGINAL SERVER
NO
106
RETURN TO CORE MODULE
YES
108
CREATE A NEW COOKIE FOR ORIGINAL SERVER USING EXTRACTED NAME AND WCOS COOKIE VALUE
110
SEND NEW COOKIE TO CORE MODULE

FIG. 4


10
12
21
CORE MODULE
23
COOKIE MODULE
15
13
CPU
INTERFACE
17
19
COMMUNICATIONS NETWORK 18(1)
COMMUNICATIONS NETWORK 18(2)
WEB SERVER DEVICE
16(n)
16(1)
14(1)
14(n)
CLIENT COMPUTING DEVICE
MEMORY
INTERFACE
DISPLAY
CPU
INPUT DEVICE
20
22
24
26
28
30

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2773088 A1 2011/03/17

(21) 2 773 088

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(86) Date de dépôt PCT/PCT Filing Date: 2010/09/09
(87) Date publication PCT/PCT Publication Date: 2011/03/17
(85) Entrée phase nationale/National Entry: 2012/03/01
(86) N° demande PCT/PCT Application No.: US 2010/048282
(87) N° publication PCT/PCT Publication No.: 2011/031868
(30) Priorité/Priority: 2009/09/10 (US12/584,697)

(51) Cl.Int./Int.Cl. *G06F 15/16* (2006.01)
(71) Demandeur/Applicant:
USABLENET INC., US

(72) Inventeur/Inventor:
SCODA, ENRICO, IT

(74) Agent: GOWLING LAFLEUR HENDERSON LLP

(54) Titre : PROCEDES D'OPTIMISATION D'INTERACTION AVEC UNE FORME DANS UNE PAGE DE SITE ET SYSTEMES ASSOCIES

(54) Title: METHODS FOR OPTIMIZING INTERACTION WITH A FORM IN A WEBSITE PAGE AND SYSTEMS THEREOF

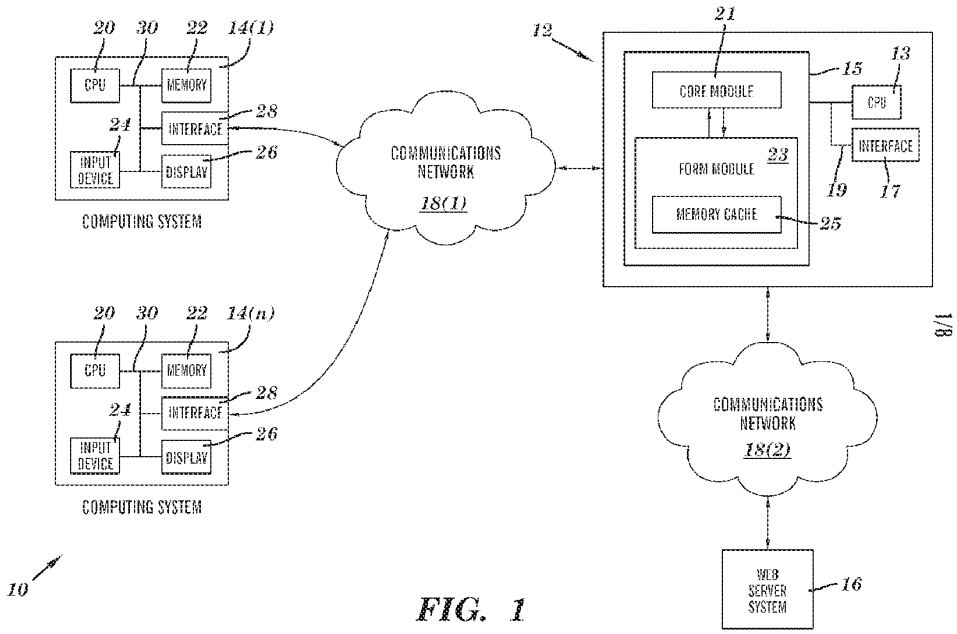


FIG. 1

(57) Abrégé/Abstract:
A method, computer readable medium and system that optimizes interaction with a form of a website page includes retrieving with a web content optimization computing device an original website page with a form in response to a request from a requesting

(57) Abrégé(suite)/Abstract(continued):
system. The web content optimization computing device stores at least the form of the original website page and adjusts at least one field control of the form to generate a simplified website page. The web content optimization computing device provides the simplified website page to the requesting system.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
17 March 2011 (17.03.2011)

PCT

(10) International Publication Number
WO 2011/031868 A1

(51) International Patent Classification:
G06F 15/16 (2006.01)

(21) International Application Number:
PCT/US2010/048282

(22) International Filing Date:
9 September 2010 (09.09.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/584,697 10 September 2009 (10.09.2009) US

(71) Applicant *(for all designated States except US)*: **US-ABLENET INC.** [US/US]; 101 Avenue Of The Americas, Suite 1204, New York, NY 10013 (US).

(72) Inventor: **SCODA, Enrico**; Via Cividina 416/3, I-33035 Martignacco (ud) (IT).

(74) Agents: **LEINBERG, Gunnar, G.** et al.; Nixon Peabody LLP-Patent Group, 1100 Clinton Square, Rochester, NY 14604-1792 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

(54) Title: METHODS FOR OPTIMIZING INTERACTION WITH A FORM IN A WEBSITE PAGE AND SYSTEMS THEREOF

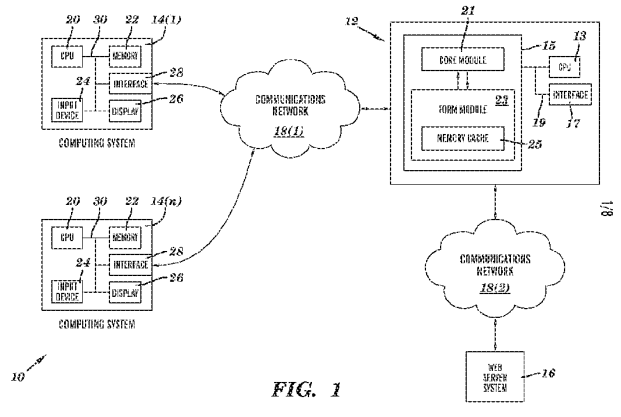


FIG. 1

(57) Abstract: A method, computer readable medium and system that optimizes interaction with a form of a website page includes retrieving with a web content optimization computing device an original website page with a form in response to a request from a requesting system. The web content optimization computing device stores at least the form of the original website page and adjusts at least one field control of the form to generate a simplified website page. The web content optimization computing device provides the simplified website page to the requesting system.

WO 2011/031868 A1

METHODS FOR OPTIMIZING INTERACTION WITH A FORM IN A WEBSITE PAGE AND SYSTEMS THEREOF

FIELD OF THE INVENTION

[0001] This application claims the benefit of U.S. Patent Application Serial No. 12/584,697, filed September 10, 2009, which is hereby incorporated by reference in its entirety.

[0002] This invention generally relates methods and systems for managing website transactions and, more particularly, methods for optimizing a form in a website page for mobile and other devices with limited memory and network capabilities.

BACKGROUND

[0003] Any website transaction, such as a hotel booking, flight check in, or concert ticket purchase, is usually broken into a series of steps which must be executed. Each step is represented by a website page containing a form a user must fill with relevant data useful to complete a task. Often these website form pages contain a number of text fields for the user to complete along with several hidden fields. Hidden fields are special website form page markup elements that contain relevant information to be transmitted between the website pages, i.e. transaction steps that must take place, but do not need to be shown to or require interaction from the user.

[0004] A web application framework is a software framework designed to support the development of dynamic websites, web applications and web services. The framework aims to alleviate overhead associated with common activities used in web development. Some of these web application frameworks keep a large amount of data inside hidden fields to save a status of the website transaction. These frameworks also automatically generate form field names which can be quite long. For example, a form field name can represent the element position inside the document by including the names of all the elements containing it.

[0005] When mobile and other devices with limited memory and network capabilities, such as smart phones and personal digital assistants, try to process a website form page built in such a manner, these devices must download a heavy weight website page, load the page and process the page in memory. When the user has entered data into the website form page, these devices also must send the website form page with the entered data along with data stored inside hidden fields back to the originating web application server for this transaction to proceed to the next step. With mobile and other devices with limited memory and network capabilities, the execution of these steps can be quite slow.

SUMMARY

[0006] A method for optimizing interaction with a form in a website page in accordance with embodiments of the present invention includes retrieving with a web content optimization computing device an original website page with a form in response to a request from a requesting system. The web content optimization computing device stores at least the form of the original website page and adjusts at least one field control of the form to generate a simplified website page. The web content optimization computing device provides the simplified website page to the requesting system.

[0007] A computer readable medium in accordance with other embodiments of the present invention includes having stored thereon instructions for optimizing interaction with a form in a website page comprising machine executable code which when executed by at least one processor, causes the processor to perform steps including retrieving an original website page with a form in response to a request. At least the form of the original website page is stored and at least one field control of the form is adjusted to generate a simplified website page. The simplified website page is provided in response to the request.

[0008] A web page form optimization system in accordance with other embodiments of the present invention includes an interface system, a storage system, and an optimization processing system in a web content optimization computing device. The interface system retrieves an original website page with a

form in response to a request from a requesting system and the storage system stores at least the form of the original website page. The optimization processing system adjusts at least one field control in the form to generate a simplified website form page. The interface system provides the simplified website page to the requesting system.

[0009] Accordingly, the present invention provides methods and systems to optimize interaction with a form in a website page with mobile and other devices with limited memory and network capabilities. The optimized form in the website page provided to these devices has substantially less markup code by removing hidden fields and shortening field control names from the form making them easier and quicker to load and process. Additionally, once fields in these optimized forms are completed with data, the resulting completed forms can more easily be transmitted back to complete the transaction step. With the present invention, these operations all take place quickly, seamlessly and without the knowledge of the end user.

BRIEF DESCRIPTION OF THE DRAWINGS

[0010] FIG. 1 is a block diagram of system in which a form of a website page is optimized in accordance with embodiments of the present invention;

[0011] FIG. 2 is a flow chart of a method for optimizing a form of a website page in accordance with embodiments of the present invention;

[0012] FIG. 3 is an exemplary form of a website page containing two text fields and a submit button;

[0013] FIG. 4A is exemplary HTML code describing the form of the website page shown in FIG. 3 with four hidden fields;

[0014] FIG. 4B is an exemplary HTTP request body resulting from the submission of the form of the website page shown in FIG. 3 and described by the code of FIG. 4A having "John" as the data entered in the first text box and "Smith" as the data entered in the second text box;

[0015] FIG. 5A is an exemplary HTML code describing the form of the website page shown in FIG. 3 after applying the method for optimizing a form of a website page in accordance with embodiments of the present invention;

[0016] FIG. 5B is an exemplary HTTP request body resulting from the submission of the form of the website page shown in FIG. 3 and described by the code of FIG. 5A having "John" as the data entered in the first text box and "Smith" as the data entered in the second text box;

[0017] FIG. 6 is a flow chart of a method for processing an HTTP request body in accordance with embodiments of the present invention; and

[0018] FIG. 7 is a diagram illustrating progressive changes to an exemplary HTTP request body when applying the method for optimizing a form of a website page in accordance with embodiments of the present invention.

DETAILED DESCRIPTION

[0019] A system 10 in which a form of a website page is optimized in accordance with embodiments of the present invention is illustrated in FIG. 1. The system 10 includes a web content optimization computing device 12, client computing devices 14(1)-14(n), a web server system 16, and communication networks 18(1)-18(2), although other numbers and types of systems, devices, and/or elements in other configurations can be used. The present invention provides a number of advantages including methods and systems to optimize interaction with website form pages when delivering content to and receiving content from mobile and other devices with limited memory and network capabilities.

[0020] Referring more specifically to FIG. 1, the web content optimization computing device 12 enables the optimization of a form of a website page, although the web content optimization computing device 12 can provide other numbers and types of functions. Although one web content optimization computing device 12 is shown, other numbers and types of web content optimization systems can be used.

[0021] The web content optimization computing device 12 includes a central processing unit (CPU) or processor 13, a memory 15, and an interface system 17 which are coupled together by a bus 19 or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor 13 in the web content optimization computing device 12 executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, including methods for optimizing interaction with website form pages, although the processor could execute other numbers and types of programmed instructions.

[0022] The memory 15 in the web content optimization computing device 12 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor 13, can be used for the memory 15 in the web content optimization computing device 12. In these embodiments, the memory 15 includes a core module 21, a form module 23, and a memory cache 25 which store programmed instructions for one or more aspects of the present invention as described and illustrated herein and some or all of original website form pages, although the memory can comprise other types and numbers of systems, devices, and elements in other configurations which store other data.

[0023] The interface system 17 in the web content optimization computing device 12 is used to operatively couple and communicate between the web content optimization computing device 12 and the client computing devices 14(1)-14(n) and the web server system 16 via the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used. By way of example only, the communication networks 18(1) and 18(2) can use TCP/IP

over Ethernet and industry-standard protocols, including HTTP, HTTPS, WAP, and SOAP, although other types and numbers of communication networks, such as a direct connection, a local area network, a wide area network, modems and phone lines, e-mail, and wireless and hardwire communication technology, each having their own communications protocols, can be used.

[0024] Each of the client computing devices 14(1)-14(n) enables a user to access content and utilize one or more applications from the web server system 16 through the web content optimization computing device 12 through one or more communication networks, although one or more of the client computing devices 14(1)-14(n) could access content and utilize other types and numbers of applications from other sources and could provide a wide variety of other functions for the user. Although multiple client computing devices 14(1)-14(n) are shown, other numbers and types of user computing systems could be used. In this example, the client computing devices 14(1)-14(n) comprise mobile devices with Internet access that permit a website form page or other retrieved data to be displayed, although each of the client computing devices 14(1)-14(n). By way of example only, one or more of the client computing devices 14(1)-14(n) can comprise smart phones, personal digital assistants, or desktop computers using visual or voice browsers.

[0025] Each of client computing devices 14(1)-14(n) includes a central processing unit (CPU) or processor 20, a memory 22, user input device 24, a display 26, and an interface system 28, and which are coupled together by a bus 30 or other link, although one or more of client computing devices 14(1)-14(n) can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 20 in each of client computing devices 14(1)-14(n) executes a program of stored instructions for one or more aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions.

[0026] The memory 22 in each of the client computing devices 14(1)-14(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the

programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor 20 can be used for the memory 22 in each of the client computing devices 14(1)-14(n).

[0027] The user input device 24 in each of the client computing devices 14(1)-14(n) is used to input selections, such as requests for a particular website form page or to enter data in fields of a form page, although the user input device could be used to input other types of data and interact with other elements. The user input device can include a keypads, touch screens, and/or vocal input processing systems although other types and numbers of user input devices can be used.

[0028] The display 26 in each of the client computing devices 14(1)-14(n) is used to show data and information to the user, such as website form page by way of example only. The display in each of the client computing devices 14(1)-14(n) is a phone screen display, although other types and numbers of displays could be used.

[0029] The interface system 28 in each of the client computing devices 14(1)-14(n) is used to operatively couple and communicate between the client computing devices 14(1)-14(n) and the web content optimization computing device 12 and web server system 16 over the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[0030] The web server system 16 provides one or more web software applications for use by one or more of the client computing devices 14(1)-14(n), although the web server system 16 can provide other numbers and types of applications and/or content and can have provide other numbers and types of functions. Although one web server system 16 is shown for ease of illustration

and discussion, other numbers and types of web server systems and devices can be used.

[0031] The web server system 16 includes a central processing unit (CPU) or processor, a memory, and an interface system which are coupled together by a bus or other link, although the web server system 16 could have other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor in the web server system 16 executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, including managing application functionality, although the processor could execute other numbers and types of programmed instructions.

[0032] The memory in the web server system 16 stores these programmed instructions for one or more aspects of the present invention as described and illustrated by way of the embodiments herein including managing application functionality, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor, can be used for the memory in each of the web server system 16.

[0033] The interface system in the web server system 16 is used to operatively couple and communicate between the web server system 16 and the web content optimization computing device 12 and the client computing devices 14(1)-14(n) via communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[0034] Although embodiments of the web content optimization computing device 12, the client computing devices 14(1)-14(n), and the web server system 16, are described and illustrated herein, each of the client computing

devices 14(1)-14(n), the web content optimization computing device 12, and the web server system 16, can be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s).

[0035] Furthermore, each of the systems of the embodiments may be conveniently implemented using one or more general purpose computer systems, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[0036] In addition, two or more computing systems or devices can be substituted for any one of the systems in any embodiment of the embodiments. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the embodiments. The embodiments may also be implemented on computer system or systems that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[0037] The embodiments may also be embodied as a computer readable medium having instructions stored thereon for one or more aspects of the present invention as described and illustrated by way of the embodiments herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the embodiments, as described and illustrated herein.

[0038] A method of optimizing a form of a website page in accordance with embodiments of the present invention will now be described with reference to FIGS. 1-5B. Referring more specifically to FIG. 2, in step 40 one of the client computing devices 14(1) transmits a Hyper Text Transfer Protocol (HTTP) request that specifies a Uniform Resource Locator (URL) to the web content optimization computing device 12 via communication network 18(1) which in turn transmits this request to the appropriate web server system 16 via communication network 18(2), although other manners for obtaining a website page can be used. The web server system 16 process the received request and transmits the requested website page back to the web content optimization computing device 12, although other types of pages and other data could be provided back in response to the request. By way of example only, a form 32 in the website page retrieved in response to this request contains a text field for Name 34, a text field for Surname 36 and a submit button 38 is illustrated in FIG. 3, although the page could have other numbers and types of fields and other elements.

[0039] In step 42, the web content optimization computing device 12 determines whether the retrieved website page has a form to optimize. If in step 42 the web content optimization computing device 12 determines the retrieved website page does not have a form to optimize, then the No branch is taken to step 50 where the retrieved website page is passed from the web content optimization computing device 12 to the requesting one of the client computing devices 14(1)-14(n). If in step 42 the web content optimization computing device 12 determines the retrieved website page does have a form to optimize, then the Yes branch is taken to step 44.

[0040] In step 44, the web content optimization computing device 12 stores code for at least the form of retrieved website page in memory, although the web content optimization computing device 12 or other devices could store other types and amounts of data, such as code for the form along with the original website page.

[0041] Additionally, in step 44, the web content optimization computing device 12 adds a new hidden field with attribute name equal to un_jtt_pid to the form of the retrieved website page, although other types and numbers of parameters or other identifiers can be added. The value attribute of un_jtt_pid hidden field control contains identification and other information to locate at least the code for the form of the retrieved website page from memory 15 of the web content optimization computing device 12.

[0042] In step 46, the web content optimization computing device 12 identifies any hidden field controls for the form, although other types of controls or other information could be identified. The web content optimization computing device 12 removes the identified hidden field controls from code for the form of the original retrieved website page, although other types and numbers of modifications to the identified hidden field controls could be executed.

~~[0043]~~ In step 48, the web content optimization computing device 12 identifies any remaining field controls for the form, although other types of controls or other information could be identified. The web content optimization computing device 12 generates a shorter name for the identified remaining field controls and replaces the initial name for each of the remaining field controls with the generated shorter name, although other types and numbers of modifications to the initial name of one or more of the remaining field controls could be executed. By way of example, in these embodiments the shorter name is generated by a '_' character followed by a number indicating the control position in the form, although other ways to generate the shorter or simpler name could be used.

[0044] Next, in step 50 if the retrieved website page had a form, then the simplified website page with the form is passed from the web content optimization computing device 12 to the requesting one of the client computing devices 14(1)-14(n).

[0045] Referring to FIG. 4A, an example of the form of the website page shown in FIG. 3 described by HTML language or code without the method for optimizing a form of a website page in accordance with embodiments of the

present invention is illustrated. The text fields 34 and 36 shown in FIG. 3 are described by long name attributes 52 and 54 in FIG. 4A. Additionally, the HTML language illustrated in FIG. 4A for the form of the original website page includes four hidden field controls 56, 58, 60 and 62. As described earlier, these hidden field controls typically are used to keep state information across multiple website pages, although they could have other functions. As illustrated in FIG. 4A, the hidden field controls 56, 58, 60 and 62 can be quite long, by way of example only some can be thousands characters long and some web applications will set the name attribute to a string containing the identification attribute of each element containing the form field control.

[0046] Referring to FIG. 4B, an exemplary HTTP request body resulting from the submission of the form of the website page shown in FIG. 3 and described by the code of FIG. 4A having "John" as the data entered in the first text box 34 in FIG. 3 and "Smith" as the data entered in the second text box 36 in FIG. 3. In these embodiments, the HTTP POST method is used to submit this data and the HTTP request body contains the encoded version of each control name attribute followed by the '=' character and the encoded version of its value attribute. Additionally, in this example the name/value pairs of each hidden field control are followed by the name value pairs of name and surname text controls.

[0047] Referring to FIG. 5A, exemplary HTML code describing the form of the website page shown in FIG. 3 after applying the method for optimizing a form of a website page in accordance with embodiments of the present invention is illustrated. As illustrated, the text fields 34 and 36 shown in FIG. 3 are now described by very short name attributes 64 and 66. Additionally, the four hidden field controls 56, 58, 60, and 62 have been removed after being saved in memory 15 in the web content optimization computing device 12. The hidden field 64 with attribute name equal to un_jtt_pid has been added. The value attribute of un_jtt_pid hidden field control contains identification and other information to locate at least the code for the form of the retrieved website page from memory 15 of the web content optimization computing device 12. Accordingly, with the present invention, the HTML code for the website page with the form is simplified

and substantially reduced which accelerates the downloading and processing of the simplified website page with the form.

[0048] Referring to FIG 5B, is an exemplary HTTP request body resulting from the submission of the form of the website page shown in FIG. 3 and described by the code of FIG. 5A having "John" as the data entered in the first text box 34 and "Smith" as the data entered the second text box 36 in FIG. 3. When the form is completed, the submit button 38 is engaged to transmit this request body with the entered data to the web content optimization device 14. As illustrated in FIG. 5B, simplifying the exemplary HTML code describing the form of the website page as illustrated in FIG. 5A by applying the method for optimizing a form of a website page in accordance with embodiments of the present invention results in a much smaller HTTP request body, than without the present invention as previously described and illustrated with respect to FIG. 4B.

[0049] Referring to FIG 6, a method for processing an HTTP request body in accordance with embodiments of the present invention is illustrated. In step 68, the web content optimization computing device 12 receives the HTTP request body with data entered, such as the one illustrated in FIG. 5B by way of example only.

[0050] In step 70, the web content optimization computing device 12 determines whether the received HTTP request body contains the un_jtt_pid parameter, although other types and numbers of identifiers could be used. If in step 70 the web content optimization computing device 12 determines the received HTTP request body does not contain the un_jtt_pid parameter, then the No branch is taken to step 78 where the received HTTP request body is transmitted to the web server system 16 or other source of the original web page. If in step 70 the web content optimization computing device 12 determines the received HTTP request body does contain the un_jtt_pid parameter, then the Yes branch is taken to step 72.

[0051] In step 72, the web content optimization computing device 12 uses the un_jtt_pid parameter to identify the original website page with the form in

memory 15 in the web content optimization computing device 12, although the information could be retrieved from other locations in other manners. Additionally, the web content optimization computing device 12 removes the un_jtt_pid parameter from the received HTTP request body.

[0052] In step 74, the web content optimization computing device 12 identifies the field controls which were renamed, retrieves the initial names from in memory 15 in the web content optimization computing device 12, and replaces the shortened or simplified names with the original corresponding initial names, although other manners for replacing some or all of the names can be used.

[0053] In step 76, the web content optimization computing device 12 identifies the hidden field controls which were removed, retrieves the hidden field controls from in memory 15 in the web content optimization computing device 12, and adds the retrieved hidden field controls back in the received HTTP request body, although other manners for reinstating some or all of this data can be used.

[0054] In step 78, when the received HTTP request body contains the un_jtt_pid parameter, the web content optimization computing device 12 transmits the received HTTP request body with the replaced names and hidden field controls to the web server system 16 or other source of the original web page.

[0055] Referring to FIG. 7, a diagram of changes to an exemplary HTTP request body when applying the method for optimizing a form of a website page in accordance with embodiments of the present invention. Block 80 illustrates the exemplary HTTP request body resulting from the submission of the original website page with the form. Block 82 illustrates the exemplary HTTP request body resulting from the submission of the original website page with the form after adding the un_jtt_pid parameter. Block 84 illustrates the exemplary HTTP request body after removing hidden field controls from the original form. Block 86 illustrates the exemplary HTTP request body after renaming remaining field controls.

[0056] Accordingly, as illustrated and described herein the present invention provides methods and systems to optimize interaction with a form in a

website page with mobile and other devices with limited memory and network capabilities. The optimized form in the website page provided to these devices has substantially less markup code by removing hidden fields and shortening field control names from the form making them easier and quicker to load and process. Additionally, once fields in these optimized forms are completed with data, the resulting completed forms can more easily be transmitted back to complete the transaction step. With the present invention, these operations all take place quickly, seamlessly and without the knowledge of the end user.

[0057] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for optimizing interaction with a form of a website page, the method comprising:

retrieving with a web content optimization computing device an original website page with a form in response to a request from a requesting system;

storing with the web content optimization computing device at least the form from the original website page;

adjusting with the web content optimization computing device at least one field control in the form to generate a simplified website page; and

providing with the web content optimization computing device the simplified website page to the requesting system.

2. The method as set forth in claim 1 wherein the adjusting further comprises:

identifying with the web content optimization computing device one or more hidden field controls in the form; and

removing with the web content optimization computing device the one or more identified hidden field controls in the form.

3. The method as set forth in claim 2 wherein the adjusting further comprises:

identifying with the web content optimization computing device one or more remaining field controls in the form; and

replacing with the web content optimization computing device an initial name of the one or more identified remaining field controls with a shorter name.

4. The method as set forth in claim 1 further comprising adding with the web content optimization computing device an identifying field

control in the simplified website page, the identifying field control providing at least an indication of the stored location of at least the form of the original website page.

5. The method as set forth in claim 1 further comprising:

receiving at the web content optimization computing device a request in response to the providing of the simplified website page;

transforming with the web content optimization computing device at least one parameter of the received request based on the at least one adjusted field control of the simplified website page;

providing with the web content optimization computing device the transformed received request to a source system of the original website page.

6. The method as set forth in claim 5 wherein the transforming further comprises:

identifying with the web content optimization computing device at least one hidden field control removed from the form; and

adding with the web content optimization computing device the at least one identified removed hidden field control as at least one additional parameter to the received request.

7. The method as set forth in claim 6 wherein the transforming further comprises:

identifying with the web content optimization computing device one or more remaining field controls in the form which were renamed; and

replacing with the web content optimization computing device the at least one parameter of the received request corresponding to the identified one or more remaining field controls which were renamed with the initial name of the one or more identified remaining field controls in the form.

8. A computer readable medium having stored thereon instructions for optimizing interaction with a form of a website page comprising

machine executable code which when executed by at least one processor, causes the processor to perform steps comprising:

retrieving with a web content optimization computing device an original website page with a form in response to a request from a requesting system;

storing with the web content optimization computing device at least the form from the original website page;

adjusting with the web content optimization computing device at least one field control in the form to generate a simplified website page; and

providing with the web content optimization computing device the simplified website page to the requesting system.

9. The medium as set forth in claim 8 wherein the adjusting further comprises:

identifying with the web content optimization computing device one or more hidden field controls in the form; and

removing with the web content optimization computing device the one or more identified hidden field controls in the form.

10. The medium as set forth in claim 9 wherein the adjusting further comprises:

identifying with the web content optimization computing device one or more remaining field controls in the form; and

replacing with the web content optimization computing device an initial name of the one or more identified remaining field controls with a shorter name.

11. The medium as set forth in claim 8 further comprising adding with the web content optimization computing device an identifying field control in the simplified website page, the identifying field control providing at least an indication of the stored location of at least the form of the original website page.

12. The medium as set forth in claim 8 further comprising:

receiving a request in response to the providing of the simplified website page;

transforming at least one parameter of the received request based on the at least one adjusted field control of the simplified website page;

providing the transformed received request to a source system of the original website page.

13. The medium as set forth in claim 12 wherein the transforming further comprises:

identifying at least one hidden field control removed from the form; and

adding the at least one identified removed hidden field control as at least one additional parameter to the received request.

14. The medium as set forth in claim 13 wherein the transforming further comprises:

identifying one or more remaining field controls which were renamed; and

replacing the at least one parameter of the received request corresponding to the identified one or more remaining field controls which were renamed with the initial name of the one or more identified remaining field controls in the form.

15. A web page form optimization system comprising:

an interface system in a web content optimization computing device that retrieves an original website page with a form in response to a request from a requesting system;

a storage system associated with the web content optimization computing device that stores at least the form from the original website page; and

an optimization processing system in the web content optimization computing device that adjusts at least one field control of the form to generate a simplified website page;

the interface system providing the simplified website page to the requesting system.

16. The system as set forth in claim 15 wherein the optimization processing system further comprises an identification processing system in the web content optimization computing device that identifies one or more hidden field controls in the form, the optimization processing system removes the one or more identified hidden field controls in the form.

17. The system as set forth in claim 16 wherein the identification system identifies one or more remaining field controls in the form and the field control processing system replaces an initial name of the one or more identified remaining field controls with a shorter name.

18. The system as set forth in claim 17 further comprising an identification system in the web content optimization computing device adds an identifying field control in the form, the identifying field control providing at least an indication of the stored location of at least the form of the original website page.

19. The system as set forth in claim 15 wherein the interface system receives a request in response to the providing of the simplified website page, the optimization processing system transforms at least one parameter of the received request based on the at least one adjusted field control of the simplified website page, and the interface system provides the transformed received request to a source system of the original website page.

20. The system as set forth in claim 19 wherein the optimization processing system further comprises an identification processing system with the web content optimization computing device that identifies at least

one hidden field control removed from the form and the optimization processing system adds at least one identified removed hidden field control as at least one additional parameter to the received request.

21. The system as set forth in claim 20 wherein the identification processing system identifies one or more remaining field controls in the form which were renamed and the optimization processing system replaces the at least one parameter of the received request corresponding to the identified one or more remaining field controls which were renamed with the initial name of the one or more identified remaining field controls.

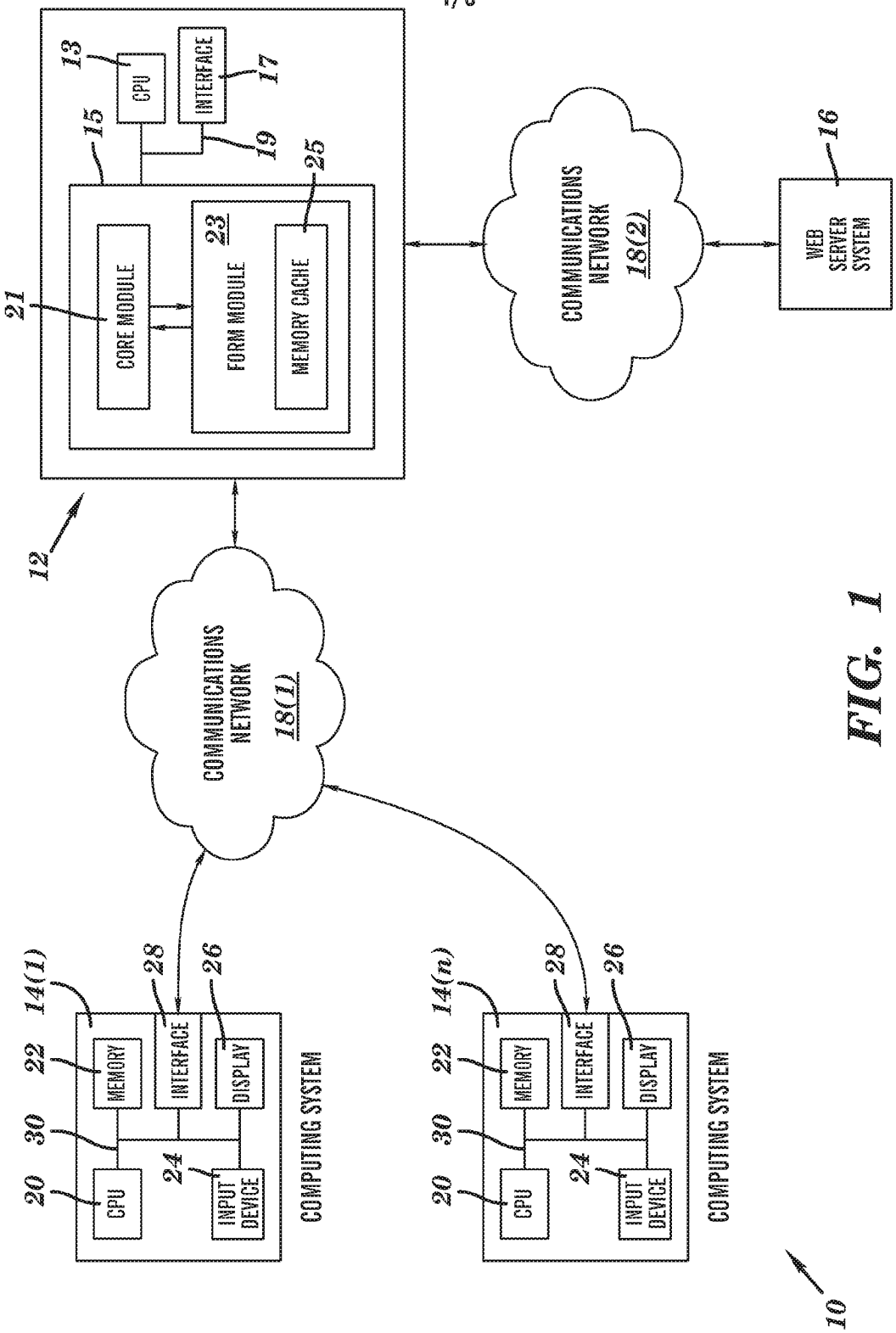

12
21
CORE MODULE
23
FORM MODULE
25
MEMORY CACHE
15
13
CPU
17
INTERFACE
19
COMMUNICATIONS NETWORK 18(1)
COMMUNICATIONS NETWORK 18(2)
16
WEB SERVER SYSTEM
14(1)
14(n)
22
MEMORY
28
INTERFACE
26
DISPLAY
30
20
CPU
24
INPUT DEVICE
COMPUTING SYSTEM
10

FIG. 1

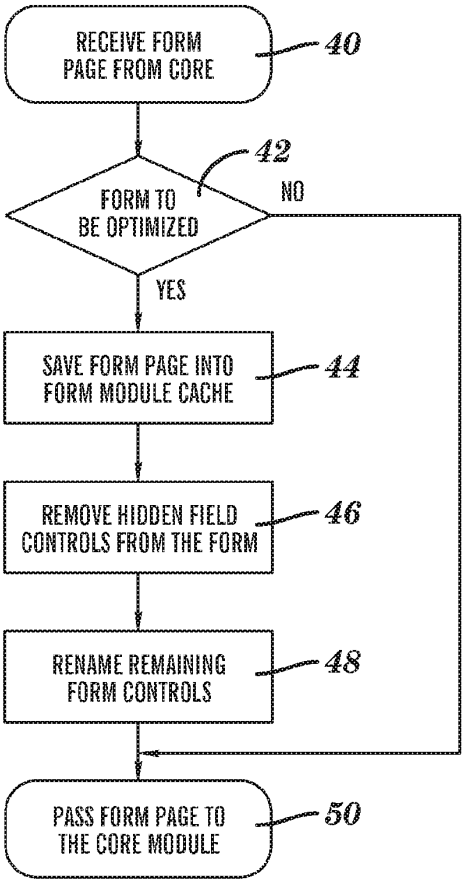

RECEIVE FORM PAGE FROM CORE
40
42
FORM TO BE OPTIMIZED
NO
YES
SAVE FORM PAGE INTO FORM MODULE CACHE
44
REMOVE HIDDEN FIELD CONTROLS FROM THE FORM
46
RENAME REMAINING FORM CONTROLS
48
PASS FORM PAGE TO THE CORE MODULE
50

FIG. 2

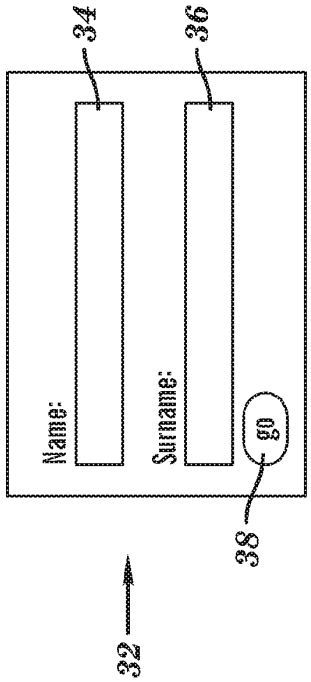


FIG. 3

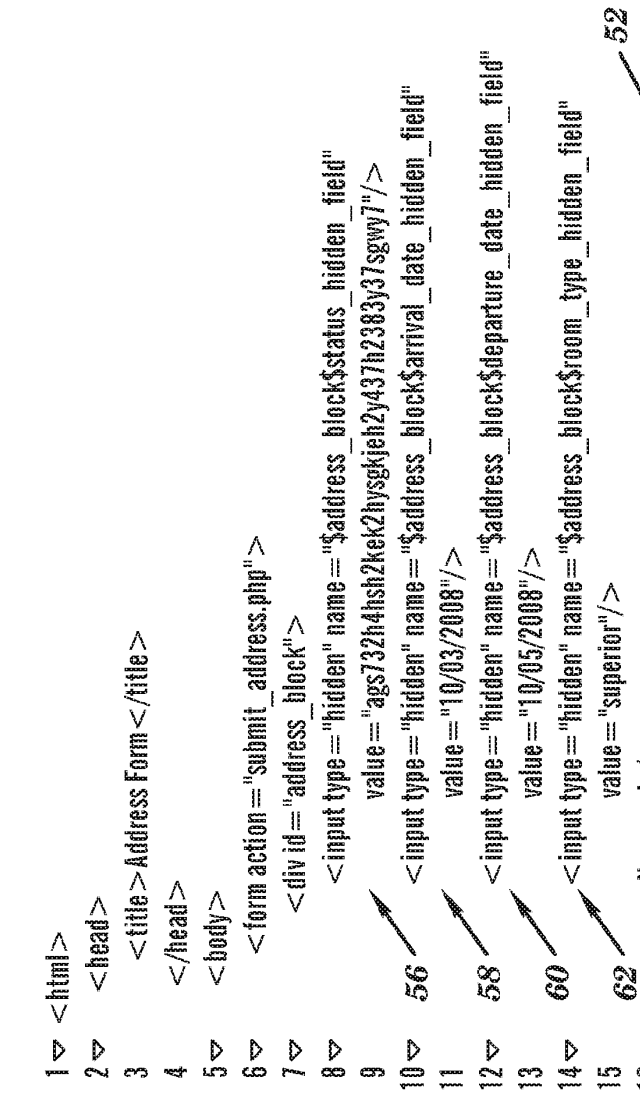

1 ▷ <html>
2 ▷ <head>
3 <title>Address Form</title>
4 </head>
5 ▷ <body>
6 ▷ <form action="submit_address.php">
7 ▷
8 ▷ <input type="hidden" name="$address_block$status_hidden_field"
9 value="ags732h4hsh2kek2hysgkjeh2y437h2383y37sgwy7"/>
10 ▷ 56 <input type="hidden" name="$address_block$arrival_date_hidden_field"
11 value="10/03/2008"/>
12 ▷ 58 <input type="hidden" name="$address_block$departure_date_hidden_field"
13 value="10/05/2008"/>
14 ▷ 60 <input type="hidden" name="$address_block$room_type_hidden_field"
15 value="superior"/>
16 62 Name:
17 <input type="text" name="$address_block$name_text_hidden_field"/>
18 Surname:
19 <input type="text" name="$address_block$surname_text_field"/>
20 <input type="submit" value="go"/>
21
22 </form>
23 </body>
24 </html>
52
54

FIG. 4A

%24address_block%24status_hidden_field=ags732h4hsh2kek2hysgkjeh2y437h2383y37sgwy7&
%24address_block%24arrival_date_hidden_field=10%2F03%2F20088&
%24address_block%24departure_date_hidden_field=10%2F05%2F20088&
%24address_block%24room_type_hidden_field=superior&
%24address_block%24name_text_field=John&%24address_block%24surname_text_field=Smith

FIG. 4B

```
1  <html>
2    <head>
3      <title>Address Form</title>
4    </head>
5    <body>
6      <form action="submit_address.php">
7        <div id="address_block">
8          Name:<br/>                                    64
9          <input type="text" name="_1"/><br/>
10         Surname:<br/>                                 66
11         <input type="text" name="_2"/><br/>
12         <input type="submit" value="go"/>
13         <input type="hidden" name="un_jtt_pid" value="n+16263748172+0"/>
14       <div/>
15     </form>
16   </body>
17 </html>
```

FIG. 5A

67

```
_1=John&_2=Smith&un_jtt_pid=n+16263748172+0
```

FIG. 5B

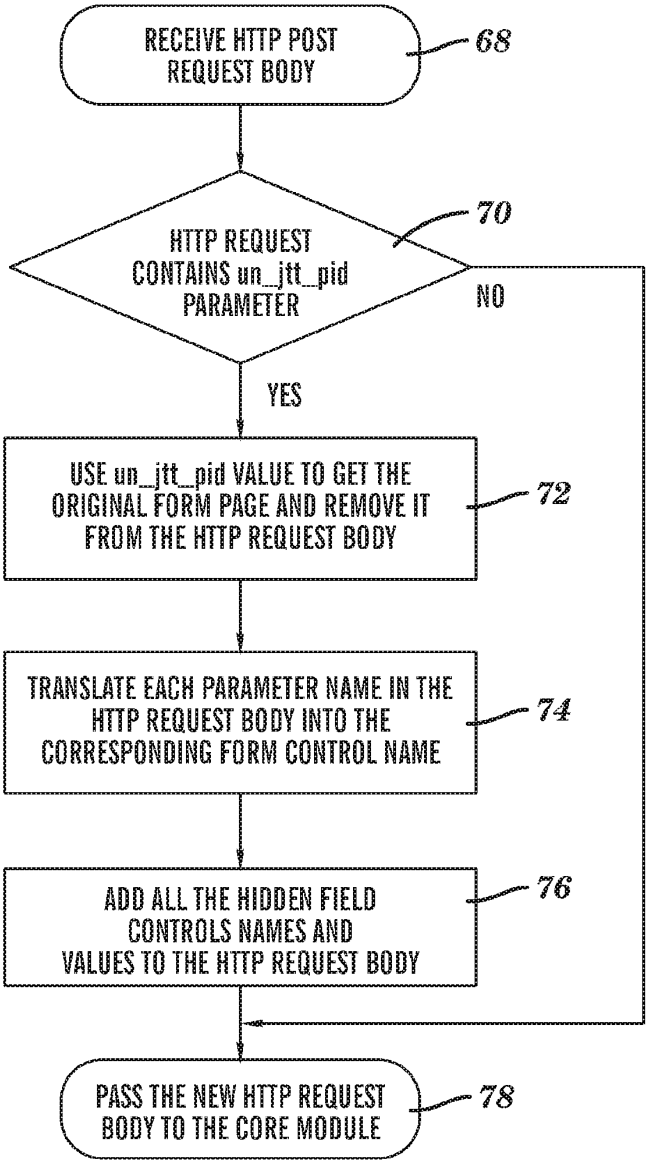

RECEIVE HTTP POST REQUEST BODY
68
HTTP REQUEST CONTAINS un_jtt_pid PARAMETER
70
NO
YES
USE un_jtt_pid VALUE TO GET THE ORIGINAL FORM PAGE AND REMOVE IT FROM THE HTTP REQUEST BODY
72
TRANSLATE EACH PARAMETER NAME IN THE HTTP REQUEST BODY INTO THE CORRESPONDING FORM CONTROL NAME
74
ADD ALL THE HIDDEN FIELD CONTROLS NAMES AND VALUES TO THE HTTP REQUEST BODY
76
PASS THE NEW HTTP REQUEST BODY TO THE CORE MODULE
78

FIG. 6

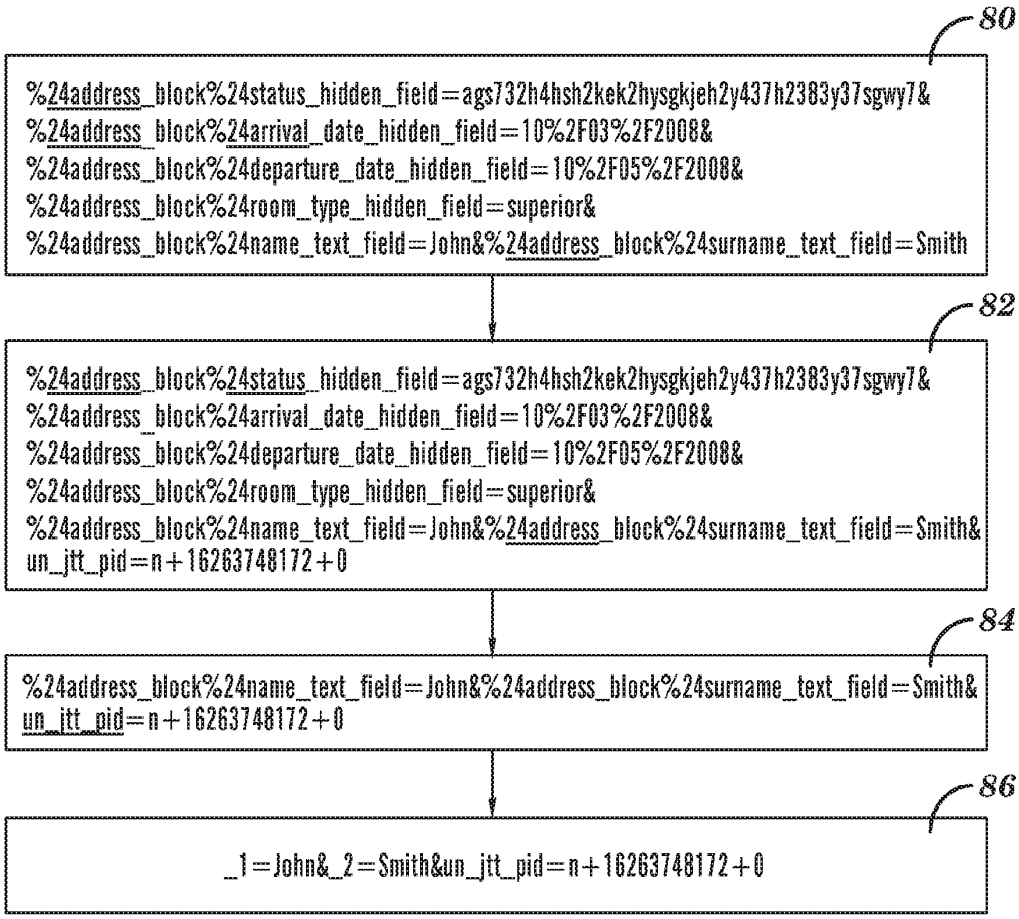

80
%24address_block%24status_hidden_field=ags732h4hsh2kek2hysgkjeh2y437h2383y37sgwy7&
%24address_block%24arrival_date_hidden_field=10%2F03%2F2008&
%24address_block%24departure_date_hidden_field=10%2F05%2F2008&
%24address_block%24room_type_hidden_field=superior&
%24address_block%24name_text_field=John&%24address_block%24surname_text_field=Smith
82
%24address_block%24status_hidden_field=ags732h4hsh2kek2hysgkjeh2y437h2383y37sgwy7&
%24address_block%24arrival_date_hidden_field=10%2F03%2F2008&
%24address_block%24departure_date_hidden_field=10%2F05%2F2008&
%24address_block%24room_type_hidden_field=superior&
%24address_block%24name_text_field=John&%24address_block%24surname_text_field=Smith&
un_jtt_pid=n+16263748172+0
84
%24address_block%24name_text_field=John&%24address_block%24surname_text_field=Smith&
un_jtt_pid=n+16263748172+0
86
_1=John&_2=Smith&un_jtt_pid=n+16263748172+0

FIG. 7

10
12
13
CPU
INTERFACE
17
15
19
25
21
CORE MODULE
23
FORM MODULE
MEMORY CACHE
COMMUNICATIONS NETWORK
18(2)
16
WEB SERVER SYSTEM
COMMUNICATIONS NETWORK
18(1)
14(1)
28
26
22
MEMORY
INTERFACE
DISPLAY
30
20
CPU
24
INPUT DEVICE
COMPUTING SYSTEM
14(n)
28
26
22
MEMORY
INTERFACE
DISPLAY
30
20
CPU
24
INPUT DEVICE
COMPUTING SYSTEM

FIG. 1

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2800723 A1 2011/12/15

(21) 2 800 723

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(86) Date de dépôt PCT/PCT Filing Date: 2011/06/10
(87) Date publication PCT/PCT Publication Date: 2011/12/15
(85) Entrée phase nationale/National Entry: 2012/11/23
(86) N° demande PCT/PCT Application No.: US 2011/040026
(87) N° publication PCT/PCT Publication No.: 2011/156743
(30) Priorité/Priority: 2010/06/11 (US12/802,670)

(51) Cl.Int./Int.Cl. *G06F 17/00* (2006.01),
G06F 9/455 (2006.01)

(71) Demandeur/Applicant:
USABLENET INC., US

(72) Inventeur/Inventor:
SCODA, ENRICO, IT

(74) Agent: GOWLING LAFLEUR HENDERSON LLP

(54) Titre : PROCEDES D'UTILISATION D'EMULATEUR JAVASCRIPT DANS SERVEUR MANDATAIRE DE CONTENU WEB ET DISPOSITIFS CORRESPONDANTS

(54) Title: METHODS FOR UTILIZING A JAVASCRIPT EMULATOR IN A WEB CONTENT PROXY SERVER AND DEVICES THEREOF

(57) Abrégé/Abstract:
A method, computer readable medium and apparatus that utilize a JavaScript emulator in a proxy server to create and store an object model of a web page which has one or more JavaScript instruction sets. At least one of the one or more JavaScript instruction sets are extracted from the web page and a JavaScript field identifier is inserted into the web page to optimize the web page which is then provided.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
15 December 2011 (15.12.2011)

PCT

(10) International Publication Number
WO 2011/156743 A3

(51) International Patent Classification:
G06F 17/00 (2006.01) *G06F 9/455* (2006.01)

(21) International Application Number:
PCT/US2011/040026

(22) International Filing Date:
10 June 2011 (10.06.2011)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/802,670 11 June 2010 (11.06.2010) US

(71) Applicant *(for all designated States except US)*: **US-ABLENET INC.** [US/US]; 28 W. 23rd Street, 6th Floor, New York, NY 10010 (US).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: **SCODA, Enrico** [IT/IT]; Via Cividina 416/3, I-33035 Matignacco (ud) (IT).

(74) Agent: **LEINBERG, Gunnar, G.**; Leclairryan, 290 Linden Oaks, Rochester, NY 14625 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(88) Date of publication of the international search report:
29 March 2012

(54) Title: METHODS FOR UTILIZING A JAVASCRIPT EMULATOR IN A WEB CONTENT PROXY SERVER AND DEVICES THEREOF

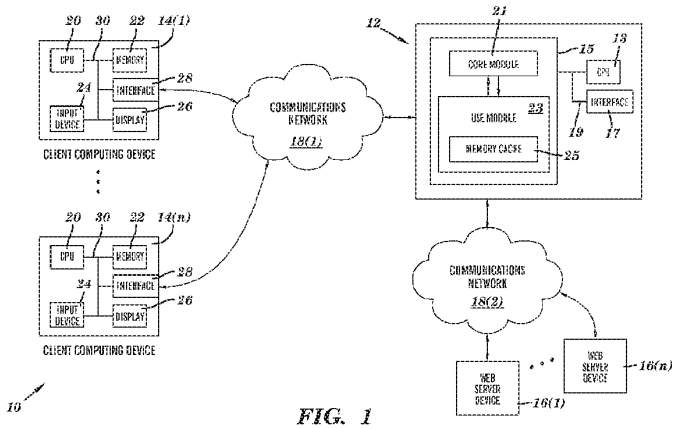


FIG. 1

(57) Abstract: A method, computer readable medium and apparatus that utilize a JavaScript emulator in a proxy server to create and store an object model of a web page which has one or more JavaScript instruction sets. At least one of the one or more JavaScript instruction sets are extracted from the web page and a JavaScript field identifier is inserted into the web page to optimize the web page which is then provided.

WO 2011/156743 A3

METHODS FOR UTILIZING A JAVASCRIPT EMULATOR IN A WEB CONTENT PROXY SERVER AND DEVICES THEREOF

FIELD

[0001] This invention generally relates to proxy servers and, more particularly, to methods for utilizing a JavaScript emulator in a web content proxy server and apparatuses thereof.

BACKGROUND

[0002] A web content optimization server is a proxy server that optimizes web page interactions for client devices with special requirements, such as mobile phones, PDAs, and smartphones and for browsing tools used by visitors with special needs, such as visual impaired users. By way of example, a web content optimization server optimizes web page interactions as follows. A client device sends an HTTP request for a web page. The web content optimization server downloads the requested original web page from the content server listed in the received request. Next, the web content optimization server optimizes the content of the web page by applying transformation rules tailored to the requesting client device. This optimization process includes extracting the content relevant to the requesting client device and adapting this extracted content to fit the specifications of the requesting client device. By way of example, these transformations include JavaScript removal, content linearization, and small screen adaptation.

[0003] Although this process works well to optimize content for display at the requesting client device, the optimization process may fail when the original content heavily depends on JavaScript technology. For example, if the requested web page includes a JavaScript code or instruction set responsible for populating form fields, validating form submissions, retrieving data from external resources (based on AJAX technology), and even generating components that may change the structure of the web page, the page at the client device will not be able to properly function.

[0004] One example of how a JavaScript instruction set operates in a web page is illustrated with reference to FIGS. 2-4. Referring more specifically to FIG. 2, an exemplary web page of a login form 100 used to gain access to a web site is illustrated. To execute the login process, a user must enter the correct a user identification into the user id field 102 and a password in the illustrated password field 104 and then click on the login button 106 to submit the login inquiry. The specific action of clicking or otherwise engaging the login button 106 triggers the execution of the JavaScript validateLogin() function or instruction set illustrated in FIG. 3. This JavaScript validateLogin() function is responsible for validating that the user id field 102 and the password field 104 were properly filled before transmitting to the web content server for the requested web site.

[0005] Referring to FIG. 4, a scenario where the user entered the value "John" into the user id field 102, but did not enter any value into the password field 104 is illustrated. When the login button 106 is clicked or otherwise engaged, the validateLogin() function generates an error and provides the message "Password cannot be empty" to the HTML element of the displayed web page 100 whose attribute id has the value "error". When the validateFunction() generates an error the data entered by the user is not sent to the web content server for the requested web site.

[0006] If, by way of example, an existing web content optimization server optimizes the content of the web page 100 illustrated in FIGS. 2 and 4 by applying transformation rules to remove the JavaScript programmed instruction set that processes any values entered in user id field 102 and password field 104 and initiates an evaluation of entered values when login button 106 is clicked, any values entered could not be processed. Additionally, any attempt to click or otherwise engage the login button 106 would not provide the desired login functionality. As a result, the web page would have been optimized for viewing on the client device, but with substantially reduced functionality.

SUMMARY

[0007] A method for utilizing a JavaScript emulator in a proxy server includes creating and storing with a proxy server an object model of a web page which has one or more JavaScript instruction sets. At least one of the one or more JavaScript instruction sets are extracted from the web page and a JavaScript field identifier is inserted into the web page with the proxy server to optimize the web page which is then provided.

[0008] A computer readable medium having stored thereon instructions for utilizing a JavaScript emulator comprising machine executable code which when executed by at least one processor, causes the processor to perform steps including creating and storing an object model of a web page which has one or more JavaScript instruction sets. At least one of the one or more JavaScript instruction sets are extracted from the web page and a JavaScript field identifier is inserted into the web page to optimize the web page which is then provided.

[0009] A web proxy apparatus includes one or more processors and a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory including creating and storing an object model of a web page which has one or more JavaScript instruction sets. At least one of the one or more JavaScript instruction sets are extracted from the web page and a JavaScript field identifier is inserted into the web page to optimize the web page which is then provided.

[00010] A method for utilizing a JavaScript emulator in a proxy server includes loading with the proxy server a stored object model of an original web page which had at least a portion of one or more JavaScript instruction sets previously extracted to form an optimized web page. Any data in a request received at the proxy server which corresponds to the loaded stored object model of the original web page is appended with the proxy server into the loaded stored object model of the original web page. The loaded stored object model of the original web page with any of the appended data is processed and then provided by the proxy server.

[00011] A computer readable medium having stored thereon instructions for utilizing a JavaScript emulator comprising machine executable code which when executed by at least one processor, causes the processor to perform steps including loading a stored object model of an original web page which had at least a portion of one or more JavaScript instruction sets previously extracted to form an optimized web page. Any data in a received request which corresponds to the loaded stored object model of the original web page is appended into the loaded stored object model of the original web page. The loaded stored object model of the original web page with any of the appended data is processed and then provided.

[00012] A web proxy apparatus includes one or more processors and a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory including loading a stored object model of an original web page which had at least a portion of one or more JavaScript instruction sets previously extracted to form an optimized web page. Any data in a received request which corresponds to the loaded stored object model of the original web page is appended into the loaded stored object model of the original web page. The loaded stored object model of the original web page with any of the appended data is processed and then provided.

[00013] A method for utilizing a JavaScript emulator in a proxy server includes creating with a proxy server an object model of a web page which has at least one JavaScript instruction set. The at least one JavaScript instruction set from the web page is executed with the proxy server to obtain content. The at least one JavaScript instruction set is removed with the proxy server from the web page and the obtained content is appended to generate an optimized web page with the obtained content which is provided by the proxy server.

[00014] A computer readable medium having stored thereon instructions for utilizing a JavaScript emulator comprising machine executable code which when executed by at least one processor, causes the processor to perform steps including creating an object model of a web page which has at least one JavaScript instruction set. The at least one JavaScript instruction set from the web

page is executed to obtain content. The at least one JavaScript instruction set is removed from the web page and the obtained content is appended to generate an optimized web page with the obtained content which is provided.

[00015] A web proxy apparatus includes one or more processors and a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory including creating an object model of a web page which has at least one JavaScript instruction set. The at least one JavaScript instruction set from the web page is executed to obtain content. The at least one JavaScript instruction set is removed from the web page and the obtained content is appended to generate an optimized web page with the obtained content which is provided.

[00016] This technology provides a number of advantages including providing a method, computer readable medium and an apparatus that further optimizes a web content proxy server through the utilization of a JavaScript emulator. With the JavaScript emulator, the web content proxy server can continue to apply optimizing transformation rules, while still enabling JavaScript functions in optimized HTTP requests, web pages, interactions with browsers, and other JavaScript events by to be executed with the JavaScript emulator. The JavaScript emulator emulates the behavior of all JavaScript objects that are supported by current web browsers and are used by web developers to access and change the components of web pages.

BRIEF DESCRIPTION OF THE DRAWINGS

[00017] FIG. 1 is a block diagram of an exemplary system environment with an optimized web content proxy server with a JavaScript emulator;

[00018] FIG. 2 is a screen shot of an exemplary login form;

[00019] FIG. 3 is an exemplary listing of JavaScript validation source code;

[00020] FIG. 4 is screen shot of an exemplary login form with an error message;

[00021] FIG. 5 is a flow chart of an example of a method for instantiating a JavaScript emulator and storing a document object model with an extracted JavaScript instruction set;

[00022] FIG. 6 is a flow chart of an example of a method for processing a received request at the optimized web content proxy server with the JavaScript emulator;

[00023] FIG. 7 is an exemplary listing JavaScript with a jQuery request; and

[00024] FIG. 8 is a flow chart of an example of a method for processing a received web page with a JavaScript instruction set at the optimized web content proxy server with a JavaScript emulator before extracting the JavaScript instruction set.

DETAILED DESCRIPTION

[00025] An exemplary environment 10 with a web content proxy server 12 with a JavaScript emulator is illustrated in FIG. 1. The exemplary environment 10 includes the web content proxy server or apparatus 12, client devices 14(1)-14(n), web server devices 16(1)-16(n), and communication networks 18(1)-18(2), although other numbers and types of systems, devices, and/or elements in other configurations and environments with other communication network topologies can be used. This technology provides a number of advantages including providing a method, computer readable medium and an apparatus that further optimizes a web content proxy server through the utilization of a JavaScript emulator.

[00026] Referring more specifically to FIG. 1, the web content proxy server 12 includes a central processing unit (CPU) or processor 13, a memory 15, and an interface system 17 which are coupled together by a bus 19 or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor 13 in the web content proxy server 12 executes a program of stored instructions one or

more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[00027] The memory 15 in the web content proxy server 12 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor 13, can be used for the memory 15 in the web content proxy server 12. In these embodiments, the memory 15 includes a core module 21 and a JavaScript emulator module 23 with a memory cache 25 which store programmed instructions and other information for one or more aspects of the present invention as described and illustrated herein, although the memory can comprise other types and numbers of systems, devices, and elements in other configurations which store other data. The JavaScript emulator module 23 includes programmed instructions and/or logic configured to as described and illustrated herein including executing JavaScript instructions extracted from optimized web pages or HTTP requests, although the JavaScript emulator module 23 can have other types and numbers of functions as described and illustrated herein.

[00028] The interface system 17 in the web content proxy server 12 is used to operatively couple and communicate between the web content proxy server 12 and the client devices 14(1)-14(n) and the web server devices 16(1)-16(n) via the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used. By way of example only, the communication networks 18(1) and 18(2) can use TCP/IP over Ethernet and industry-standard protocols, including HTTP, HTTPS, WAP, and SOAP, although other types and numbers of communication networks, such as a direct connection, a local area

network, a wide area network, modems and phone lines, e-mail, and wireless and hardwire communication technology, each having their own communications protocols, can be used.

[00029] Each of the client devices 14(1)-14(n) enables a user to request, get and interact with web pages from one or more web sites hosted by the web server devices 16(1)-16(n) through the web content proxy server 12 via one or more communication networks, although one or more of the client devices 14(1)-14(n) could access content and utilize other types and numbers of applications from other sources and could provide a wide variety of other functions for the user. Although multiple client devices 14(1)-14(n) are shown, other numbers and types of user computing systems could be used. In this example, the client devices 14(1)-14(n) comprise mobile devices with Internet access that permit a website form page or other retrieved data to be displayed, although each of the client devices 14(1)-14(n). By way of example only, one or more of the client devices 14(1)-14(n) can comprise smart phones, personal digital assistants, or computers.

[00030] Each of client devices 14(1)-14(n) in this example is a computing device that includes a central processing unit (CPU) or processor 20, a memory 22, user input device 24, a display 26, and an interface system 28, and which are coupled together by a bus 30 or other link, although one or more of client devices 14(1)-14(n) can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 20 in each of client devices 14(1)-14(n) executes a program of stored instructions for one or more aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions.

[00031] The memory 22 in each of the client devices 14(1)-14(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, or

other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor 20 can be used for the memory 22 in each of the client devices 14(1)-14(n.

[00032] The user input device 24 in each of the client devices 14(1)-14(n) is used to input selections, such as requests for a particular website form page or to enter data in fields of a form page, although the user input device could be used to input other types of data and interact with other elements. The user input device can include keypads, touch screens, and/or vocal input processing systems although other types and numbers of user input devices can be used.

[00033] The display 26 in each of the client devices 14(1)-14(n) is used to show data and information to the user, such as website page by way of example only. The display in each of the client devices 14(1)-14(n) is a phone screen display, although other types and numbers of displays could be used depending on the particular type of client device.

[00034] The interface system 28 in each of the client devices 14(1)-14(n) is used to operatively couple and communicate between the client devices 14(1)-14(n) and the web content proxy server 12 and web server devices 16(1)-16(n) over the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00035] The web server devices 16(1)-16(n) provide one or more pages from one or more web sites for use by one or more of the client devices 14(1)-14(n) via the web content proxy server 12, although the web server devices 16(1)-16(n) can provide other numbers and types of applications and/or content and can have provide other numbers and types of functions. Although web server devices 16(1)-16(n) are shown for ease of illustration and discussion, other numbers and types of web server systems and devices can be used.

[00036] Each of the web server devices 16(1)-16(n) include a central processing unit (CPU) or processor, a memory, and an interface system which are

coupled together by a bus or other link, although each of the web server devices 16(1)-16(n) could have other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor in each of the web server devices 16(1)-16(n) executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[00037] The memory in each of the web server devices 16(1)-16(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated by way of the embodiments, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor, can be used for the memory in each of the web server devices 16(1)-16(n).

[00038] The interface system in each of the web server devices 16(1)-16(n) is used to operatively couple and communicate between the web server devices 16(1)-16(n) and the web content proxy server 12 and the client devices 14(1)-14(n) via communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00039] Although embodiments of the web content proxy server 12, the client devices 14(1)-14(n), and the web server devices 16(1)-16(n), are described and illustrated herein, each of the client devices 14(1)-14(n), the web content proxy server 12, and the web server devices 16(1)-16(n), can be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to

implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s).

[00040] Furthermore, each of the systems of the embodiments may be conveniently implemented using one or more general purpose computer systems, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[00041] In addition, two or more computing systems or devices can be substituted for any one of the systems in any embodiment of the embodiments. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the embodiments. The embodiments may also be implemented on computer system or systems that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[00042] The embodiments may also be embodied as a computer readable medium having instructions stored thereon for one or more aspects of the present invention as described and illustrated by way of the embodiments herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the embodiments, as described and illustrated herein.

[00043] An exemplary method for utilizing a JavaScript emulator in a web content proxy server 12 in an exemplary environment 10 will now be described with reference to FIGS. 1-8. Referring more specifically to FIGS. 1 and 5, in this

example in step 202, a form web page is received at the web content proxy server 12.

[00044] In step 204, the web content proxy server 12 determines whether the received web form page has any JavaScript code or instruction sets for execution during use of the web form page, such as for one or more forms or other fields in the received form web page. If in step 204 the web content proxy server 12 determines the received web form page does not have any JavaScript programmed instruction sets, then the No branch is taken to step 214 described below. If in step 204 the web content proxy server 12 determines the received web form page does has one or more JavaScript programmed instruction sets, then the Yes branch is taken to step 206.

[00045] In step 206, the web content proxy server 12 instantiates the received web form page with the JavaScript module 23 which is responsible for emulating the behavior of each of the identified JavaScript instruction sets. During the instantiation, the web content proxy server 12 creates a document object model of the web form page with each of the identified JavaScript instruction sets, although the other types of models could be created.

[00046] In this example, for reasons of security and performance four levels of detail can be used when the web content proxy server 12 creates the document object model, although other levels of detail can be used. In this particular example, the levels of detail for the document object model are: basic, attributes, all-but-scripts, and all, although other levels of detail in the document model object can be used. With basic, during the instantiation process when creating the document object model the JavaScript module 23 includes all elements in the web page removing their content, except for the elements of a form, such as checkboxes, radio boxes, combo boxes, buttons, and text fields by way of example. The JavaScript module 23 also includes all attributes of form controls along with only the id and class attributes of all elements. With attributes, during the instantiation process when creating the document object model the JavaScript module 23 includes all elements removing their content, except for the elements of a form, such as checkboxes, radio boxes, combo boxes,

buttons, and text fields by way of example. The JavaScript module 23 also includes all attributes of form controls. With all-but-scripts, during the instantiation process when creating the document object model the JavaScript module 23 includes all elements, including both content and attributes, however the JavaScript module 23 removes the content of the elements style and script. With all, during the instantiation process when creating the document object model the JavaScript module 23 includes all elements including both content and attributes.

[00047] Once the instantiation process is completed, in step 210 the JavaScript module 23 stores the created document object model for the web page with each of the JavaScript instruction sets in the cache memory 15 and assigns a JavaScript identifier to it, although each of the created document object models could be stored in other manners and in other locations. For security reasons the JavaScript module 23 executed by the web content proxy server 12 also provides the functionality to delete any values in fields before storing the created document object model. Accordingly, the web content proxy server 12 does not store sensitive data, such as credit card numbers, either temporary or permanently.

[00048] In step 212, the JavaScript module 23 executed by the web content proxy server 12 adds the assigned JavaScript field identifier for the created document object model in the optimized web form page at each location of an extracted JavaScript instruction set. In this particular example, the JavaScript identifier for the stored document object model is called un_jtt_jse and is stored as a hidden identifier in the optimized web page form provided to the requesting one of the client devices 14(1)-14(n), although other types and numbers of identifiers which are hidden or visible could be used.

[00049] In step 214, the JavaScript module 23 passes the optimized web form page to the core module 21 to be provided by the web content proxy server 12 for further use in the exemplary environment 10, such as in a web browser on one of the client devices 14(1)-14(n).

[00050] In another illustrative example, an exemplary method for processing a received request at the optimized web content proxy server 12 with the JavaScript emulator is illustrated with reference to FIGS. 1 and 6. In step 300, the JavaScript module 23 in web content proxy server 12 receives an HTTP request from the core module 21 that was received from one of the client devices 14(1)-14(n), although other types of requests, web pages, browser interactions or other types of JavaScript related actions could be received.

[00051] In step 302, the JavaScript module 23 in the web content proxy server 12 is executed to determine if the received HTTP request includes one or more hidden JavaScript identifiers or other marker. In this particular example, an un_jtt_jse JavaScript identifier is included in the received HTTP request. If in step 302, the received HTTP request does not include one or more hidden JavaScript identifiers, then the No branch is taken to step 310 as explained below. If in step 302, the received HTTP request does include one or more hidden JavaScript identifiers, then the Yes branch is taken to step 304.

[00052] In step 304, the web content proxy server 12 executes the JavaScript module 23 to retrieve from memory cache 25 the corresponding stored document object model for the identified JavaScript identifier. In this particular example, the document object model associated with the JavaScript identifier un_jtt_jse is retrieved.

[00053] In step 306, the web content proxy server 12 executes the JavaScript module 23 to validate all of the JavaScript instruction sets in the retrieved document object model.

[00054] In step 308, the web content proxy server 12 executes with the JavaScript module 23 each of the JavaScript instruction sets in the document object model with any corresponding values in the received HTTP request. By emulating the typical behavior of a web browser, the JavaScript module 23 executed by the web content proxy server 12 also assigns default values to all those properties whose values can neither be inferred from the received HTTP

request nor from the document object module. The web content proxy server 12 appends the output of this execution to the received HTTP request.

[00055] In step 310, the JavaScript module 23 is executed by the web content proxy server 12 to pass the core module 21 the response with the executed JavaScript functionality appended to the HTTP request. The core module 21 is executed by the web content proxy server 12 to determine where to provide the response with the executed JavaScript functionality in the exemplary environment 10, such as to a web browser on one of the client devices 14(1)-14(n). For example, the response to the received HTTP request might be an error message if incorrect or insufficient values were provided or might provide other information if all the needed values were provided.

[00056] By way of example now, consider the web page form represented in FIG. 2 that can be validated by the JavaScript instruction set or code shown in FIG. 3. The action of clicking on the login button 106 of the optimized form web page 100 triggers a new HTTP request that is sent to the web content proxy server 12 which executes the programmed instructions in the JavaScript module 23. In this example, the received request would includes an un_jtt_jse identifier or parameter so the 61 then the corresponding document object model is loaded from the memory cache 25. The JavaScript module 23 is executed by the web content proxy server 12 to use the form name or its index, i.e. position within the web page or document, to bind the parameters from the received HTTP request with the corresponding controls in the stored document object model representation of the previously extracted JavaScript instruction set during optimization.

[00057] At this point, the JavaScript module 23 is executed by the web content proxy server 12 to validate the received HTTP request with the corresponding controls in the stored document object model. Next, the JavaScript module 23 is executed by the web content proxy server 12 to append 64 the output of the validation process to the data included in the received HTTP request. The JavaScript module 23 also is executed by the web content proxy server 12 to retrieve values from JavaScript variables, properties of HTML elements, such as

the innerHTML property, input.value or document.location), or the strings sent to the window.alert() and the document.write() functions as necessary.

[00058] Once the HTTP request has been modified the JavaScript module 23 sends it to the core module 21. Based on the computed data, the core module 21 is executed by the web content proxy server 12 to determine whether to print an error message or send the data from the form to the original web content server.

[00059] In an another illustrative example, an exemplary method a method for processing a received web page at the optimized web content proxy server 12 with a JavaScript emulator for delivery to one of the client computing devices 10 is illustrated with reference to FIGS. 1, 7, and 8. In step 400, the JavaScript module 23 in web content proxy server 12 receives a web page which contains a JavaScript instruction set that creates dynamic content from one of the web server devices 16(1)-16(n), although other types of requests, web pages, browser interactions or other types of JavaScript related actions could be received and the content can be static or dynamic. By way of example, consider a web page with a JavaScript code or instruction set based on the jQuery library shown in FIG. 7. This is a JavaScript framework used to develop complex web applications based on Ajax technology. The event of receiving or loading this web page triggers the execution of the JavaScript code or instruction set shown in the example.

[00060] In step 402, the web content proxy server 12 instantiates this received web page which contains the JavaScript instruction set to create a document object model of the web page.

[00061] In step 404, the web content proxy server 12 adds the JavaScript programmed instruction set to retrieve dynamic content to the created document object model. In the example illustrated in FIG. 7, the web content proxy server 12 adds the JavaScript module 23 to retrieve the content of '/cart.html' web page and place it into the html element identified by the attribute id with value "cart".

[00062] In step 406, the web content proxy server 12 executes the JavaScript programmed instruction set in the document object model of the received web page to obtain the dynamic content. For example, when executing

the JavaScript programmed instruction set in the document object model, the JavaScript module 23 can dispatch Ajax calls to retrieve content from external sources, such as one of the web server devices 16(1)-16(n). In this example, the JavaScript programmed instruction set is programmed to retrieve dynamic content, although other types of dynamic or static data from other sources could be obtained.

[00063] In step 408, the web content proxy server 12 executes the JavaScript module 23 to append the obtained content in an optimized version of the received web page with the JavaScript programmed instruction set extracted.

[00064] In step 410, the JavaScript module 23 is executed by the web content proxy server 12 passes the optimized web page with the extracted JavaScript instruction set and appended dynamic content to the core module 21. The core module 21 is executed by the web content proxy server 12 to determine where to provide the web page with the executed JavaScript functionality in the exemplary environment 10, such as to a web browser on one of the client devices 14(1)-14(n), although other manners for passing the optimized web page with the extracted JavaScript instruction set and appended dynamic content could be used.

[00065] Accordingly, as illustrated and described herein this technology provides a number of advantages including providing a method, computer readable medium and an apparatus that further optimizes a web content proxy server through the utilization of a JavaScript emulator. The processes illustrated above are only exemplary and all JavaScript functionality can be emulated by the web content proxy server 12 with this technology. With the JavaScript emulator, the web content proxy server can continue to apply optimizing transformation rules, while still enabling JavaScript functions in optimized HTTP requests, web pages, interactions with browsers, and other JavaScript events by to be executed with the JavaScript emulator. The JavaScript emulator emulates the behavior of all JavaScript objects that are supported by current web browsers and are used by web developers to access and change the components of web pages.

[00066] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for utilizing a JavaScript emulator in a proxy server, the method comprising:

creating and storing with a proxy server an object model of a web page which has one or more JavaScript instruction sets;

extracting with the proxy server at least one of the one or more JavaScript instruction sets from the web page and inserting a JavaScript field identifier into the web page to optimize the web page; and

providing with the proxy server the optimized web page with the inserted JavaScript field identifier.

2. The method as set forth in claim 1 wherein the creating and storing further comprises creating and storing with the proxy server the object model of the web page including all attributes and content of controls for one or more forms in the web page and all remaining elements of the web page with all content removed.

3. The method as set forth in claim 2 wherein the creating and storing further comprises creating and storing with the proxy server the object model of the web page with all attributes of identification and class for all the elements of the web page.

4. The method as set forth in claim 2 wherein the creating and storing with the proxy server the object model of the web page further comprises creating and storing the object model of the web page with all attributes of all the elements in the web page.

5. The method as set forth in claim 1 wherein the creating and storing further comprises creating and storing with the proxy server the object model of the web page with all elements and content retained and having elements of style and script removed.

6. The method as set forth in claim 1 wherein the extracting further comprises extracting all of the one or more JavaScript instruction sets.

7. The method as set forth in claim 1 further comprising deleting with the proxy server any values in form controls in the created object model.

8. A computer readable medium having stored thereon instructions for utilizing a JavaScript emulator comprising machine executable code which when executed by at least one processor, causes the processor to perform steps:

creating and storing an object model of a web page which has one or more JavaScript instruction sets;

extracting at least one of the one or more JavaScript instruction sets from the web page and inserting a JavaScript field identifier into the web page to optimize the web page; and

providing the optimized web page with the inserted JavaScript field identifier.

9. The medium as set forth in claim 8 wherein the creating and storing further comprises creating and storing the object model of the web page including all attributes and content of controls for one or more forms in the web page and all remaining elements of the web page with all content removed.

10. The medium as set forth in claim 9 wherein the creating and storing further comprises creating and storing the object model of the web page with all attributes of identification and class for all the elements of the web page.

11. The medium as set forth in claim 9 wherein the creating and storing the object model of the web page further comprises creating and storing the object model of the web page with all attributes of all the elements in the web page.

12. The medium as set forth in claim 8 wherein the creating and storing further comprises creating and storing the object model of the web page with all elements and content retained and having elements of style and script removed.

13. The medium as set forth in claim 8 wherein the extracting further comprises extracting all of the one or more JavaScript instruction sets.

14. The medium as set forth in claim 8 further comprising deleting any values in form controls in the created object model.

15. A web proxy apparatus comprising:

one or more processors;

a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory comprising:

creating and storing an object model of a web page which has one or more JavaScript instruction sets;

extracting at least one of the one or more JavaScript instruction sets from the web page and inserting a JavaScript field identifier into the web page to optimize the web page; and

providing the optimized web page with the inserted JavaScript field identifier.

16. The apparatus as set forth in claim 15 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for the creating and storing further comprises creating and storing the object model of the web page including all attributes and content of controls for one or more forms in the web page and all remaining elements of the web page with all content removed.

17. The apparatus as set forth in claim 16 wherein the one or more processors is further configured to execute programmed instructions stored

in the memory for the creating and storing further comprises creating and storing the object model of the web page with all attributes of identification and class for all the elements of the web page.

18. The apparatus as set forth in claim 16 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for the creating and storing further comprises creating and storing the object model of the web page with all attributes of all the elements in the web page.

19. The apparatus as set forth in claim 15 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for the creating and storing further comprises creating and storing the object model of the web page with all elements and content retained and having elements of style and script removed.

20. The apparatus as set forth in claim 15 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for the extracting further comprises extracting all of the one or more JavaScript instruction sets.

21. The apparatus as set forth in claim 15 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising deleting any values in form controls in the created object model.

22. A method for utilizing a JavaScript emulator in a proxy server, the method comprising:

loading with the proxy server a stored object model of an original web page which had at least a portion of one or more JavaScript instruction sets previously extracted to form an optimized web page;

appending with the proxy server any data in a request received at the proxy server which corresponds to the loaded stored object model

of the original web page into the loaded stored object model of the original web page;

processing with the proxy server the loaded stored object model of the original web page with any of the appended data; and

providing with the proxy server the processed loaded stored object model of the original web page.

23. The method as set forth in claim 22 further comprising determining with the proxy server whether the request corresponds to the stored object model of an original web page, wherein the loading and the appending are executed when the determining determines the request corresponds to the stored object model of an original web page.

24. The method as set forth in claim 22 wherein the processing further comprises validating with the proxy server the loaded stored object model of the original web page with any of the appended data.

25. The method as set forth in claim 22 wherein the providing further comprises providing one or more functionalities of one or more resulting properties from the processing of the loaded stored object model of the original web page with any of the appended data to a determined destination.

26. A computer readable medium having stored thereon instructions for utilizing a JavaScript emulator comprising machine executable code which when executed by at least one processor, causes the processor to perform steps:

loading a stored object model of an original web page which had at least a portion of one or more JavaScript instruction sets previously extracted to form an optimized web page;

appending any data in a request received at the proxy server which corresponds to the loaded stored object model of the original web page into the loaded stored object model of the original web page;

processing the loaded stored object model of the original web page with any of the appended data; and

providing the processed loaded stored object model of the original web page.

27. The medium as set forth in claim 26 further comprising determining whether the request corresponds to the stored object model of an original web page, wherein the loading and the appending are executed when the determining determines the request corresponds to the stored object model of an original web page.

28. The medium as set forth in claim 26 wherein the processing further comprises validating the loaded stored object model of the original web page with any of the appended data.

29. The medium as set forth in claim 26 wherein the providing further comprises providing one or more functionalities of one or more resulting properties from the processing of the loaded stored object model of the original web page with any of the appended data to a determined destination.

30. A web proxy apparatus comprising:

one or more processors;

a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory comprising:

loading a stored object model of an original web page which had at least a portion of one or more JavaScript instruction sets previously extracted to form an optimized web page;

appending any data in a request received at the proxy server which corresponds to the loaded stored object model of the original web page into the loaded stored object model of the original web page;

processing the loaded stored object model of the original web page with any of the appended data; and

providing the processed loaded stored object model of the original web page.

31. The apparatus as set forth in claim 30 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising determining whether the request corresponds to the stored object model of an original web page, wherein the loading and the appending are executed when the determining determines the request corresponds to the stored object model of an original web page.

32. The apparatus as set forth in claim 30 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for the processing further comprising validating the loaded stored object model of the original web page with any of the appended data.

33. The apparatus as set forth in claim 30 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for the providing further comprising providing one or more functionalities of one or more resulting properties from the processing of the loaded stored object model of the original web page with any of the appended data to a determined destination.

34. A method for utilizing a JavaScript emulator in a proxy server, the method comprising:

creating with a proxy server an object model of a web page which has at least one JavaScript instruction set;

executing with the proxy server the at least one JavaScript instruction set from the web page to obtain content;

removing with the proxy server the at least one JavaScript instruction set from the web page and appending the obtained content to generate an optimized web page with the obtained content; and

providing with the proxy server the optimized web page with the obtained content.

35. A computer readable medium having stored thereon instructions for utilizing a JavaScript emulator comprising machine executable code which when executed by at least one processor, causes the processor to perform steps:

creating an object model of a web page which has at least one JavaScript instruction set;

executing the at least one JavaScript instruction set from the web page to obtain content;

removing the at least one JavaScript instruction set from the web page and appending the obtained content to generate an optimized web page with the obtained content; and

providing the optimized web page with the obtained content.

36. A web proxy apparatus comprising:

one or more processors;

a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory comprising:

creating an object model of a web page which has at least one JavaScript instruction set;

executing the at least one JavaScript instruction set from the web page to obtain content;

removing the at least one JavaScript instruction set from the web page and appending the obtained content to generate an optimized web page with the obtained content; and

providing the optimized web page with the obtained content.

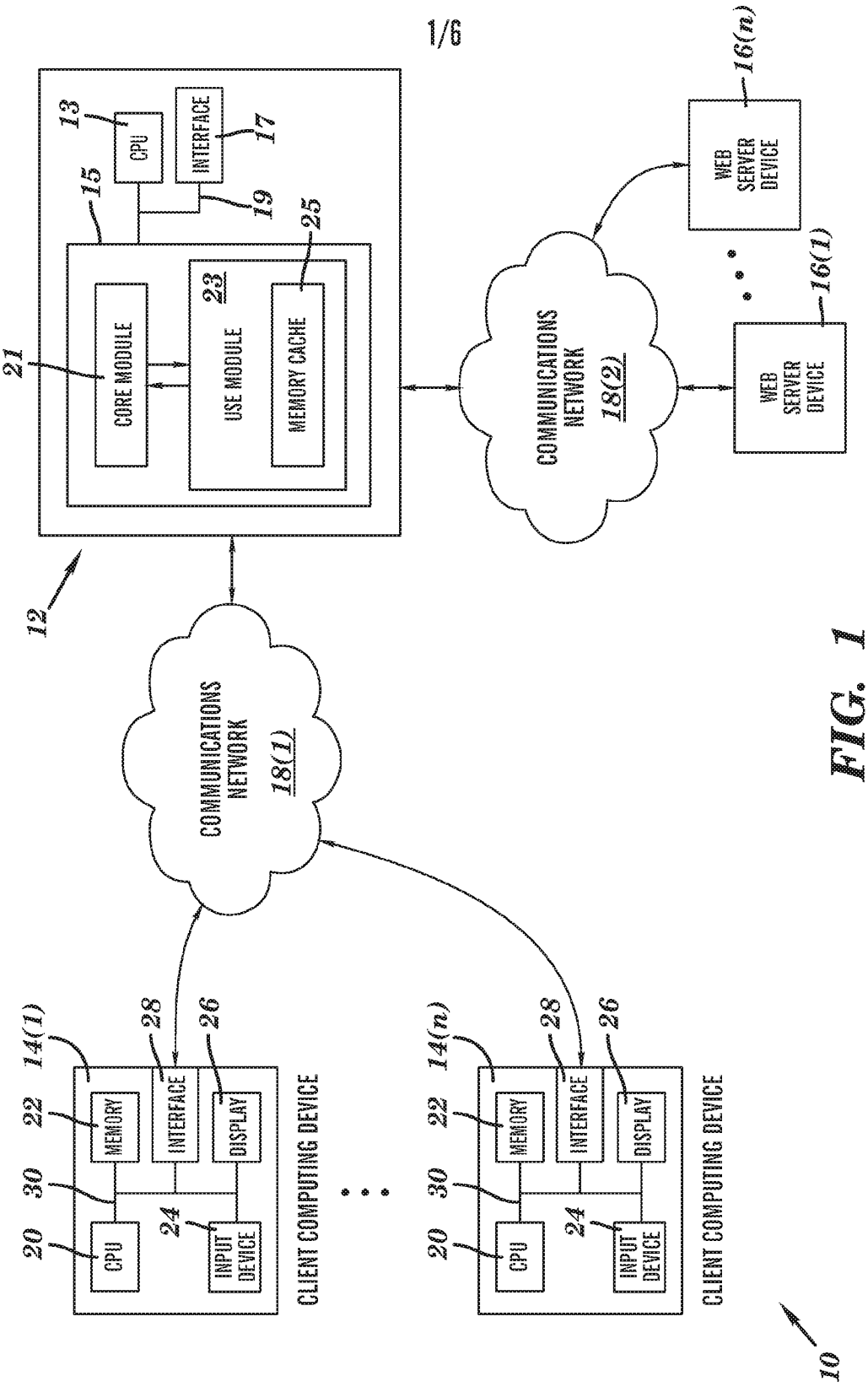

10
12
13
CPU
INTERFACE
17
15
19
25
21
CORE MODULE
23
USE MODULE
MEMORY CACHE
COMMUNICATIONS NETWORK 18(1)
COMMUNICATIONS NETWORK 18(2)
WEB SERVER DEVICE
16(n)
16(1)
14(1)
14(n)
22
MEMORY
28
INTERFACE
26
DISPLAY
30
20
CPU
24
INPUT DEVICE
CLIENT COMPUTING DEVICE

FIG. 1

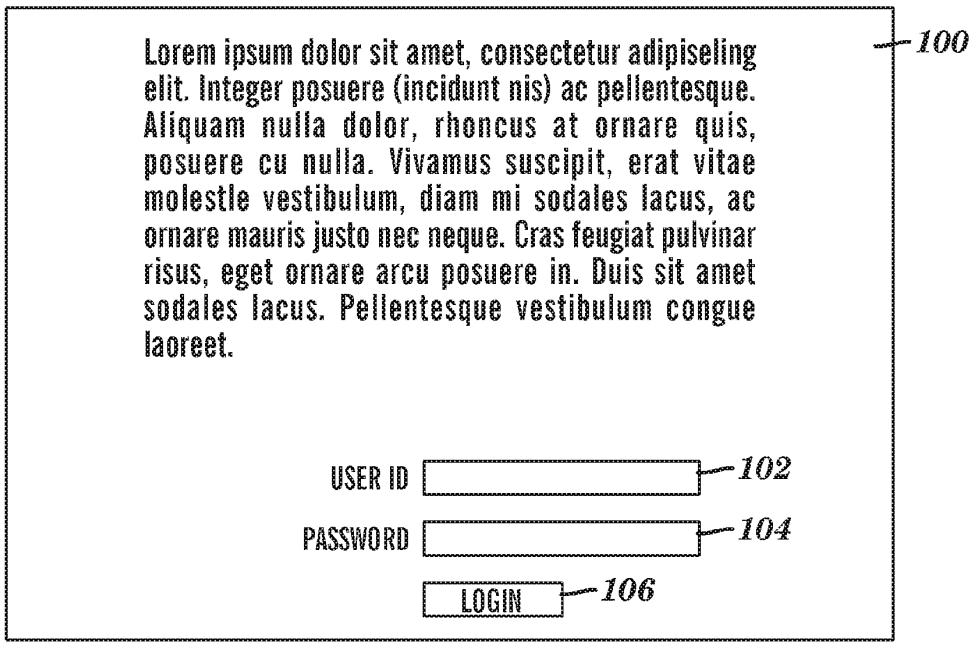


FIG. 2

```
function validateLogin() {
        var f = document.LoginForm;
        var error = document.getElementById('error');
        if (f.userId.value == '') {
                error.innerHTML = "USER ID cannot be empty";
                return false;
        }
        if (f.password.value == '') {
                error.innerHTML = "PASSWORD cannot be empty";
                return false;
        }
        return true;
}
```

FIG. 3

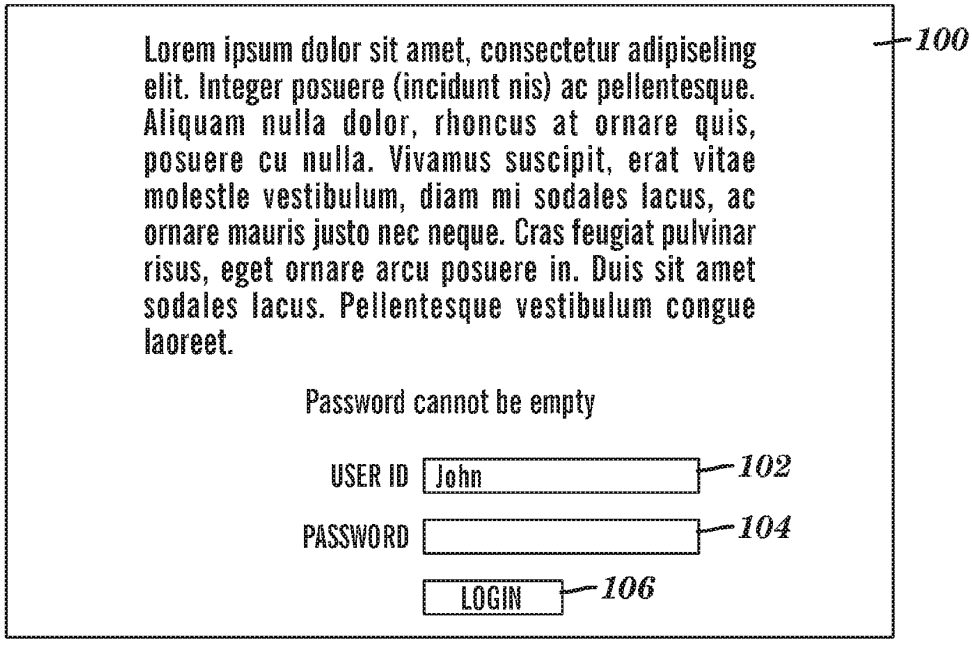

100
Lorem ipsum dolor sit amet, consectetur adipiseling elit. Integer posuere (incidunt nis) ac pellentesque. Aliquam nulla dolor, rhoncus at ornare quis, posuere cu nulla. Vivamus suscipit, erat vitae molestie vestibulum, diam mi sodales lacus, ac ornare mauris justo nec neque. Cras feugiat pulvinar risus, eget ornare arcu posuere in. Duis sit amet sodales lacus. Pellentesque vestibulum congue laoreet.
Password cannot be empty
USER ID
John
102
PASSWORD
104
LOGIN
106

FIG. 4

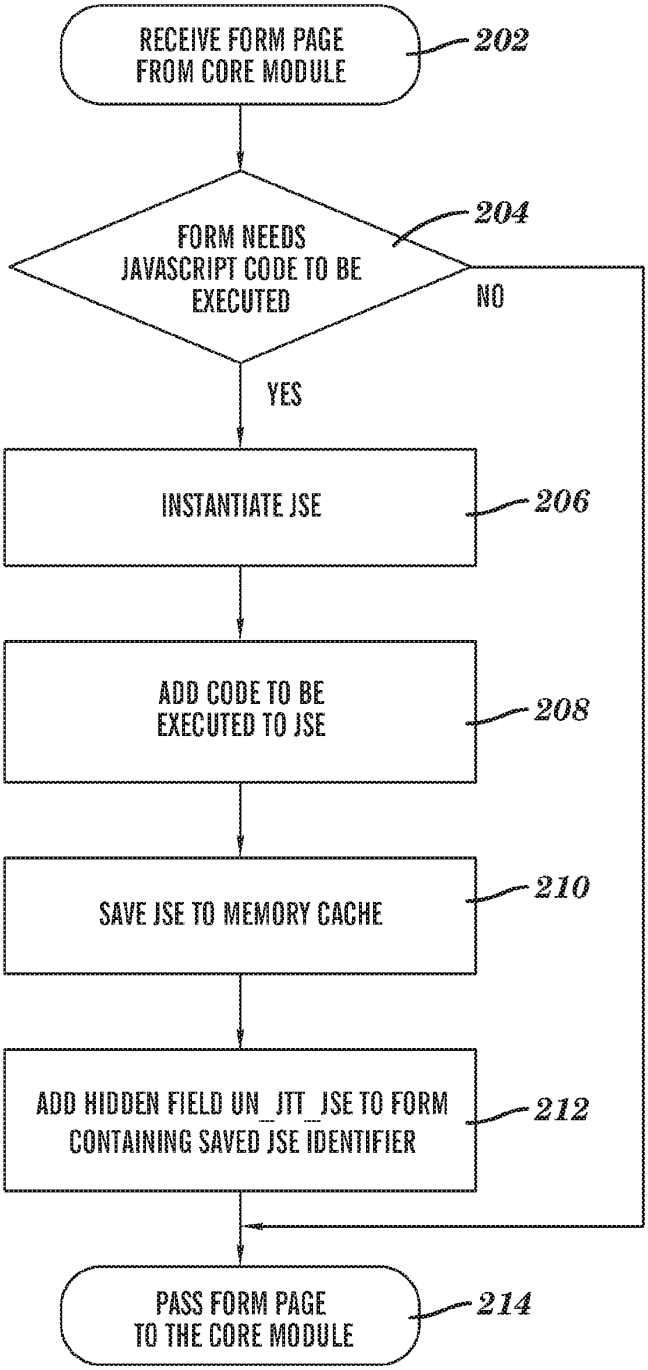

RECEIVE FORM PAGE FROM CORE MODULE
202
FORM NEEDS JAVASCRIPT CODE TO BE EXECUTED
204
NO
YES
INSTANTIATE JSE
206
ADD CODE TO BE EXECUTED TO JSE
208
SAVE JSE TO MEMORY CACHE
210
ADD HIDDEN FIELD UN_JTT_JSE TO FORM CONTAINING SAVED JSE IDENTIFIER
212
PASS FORM PAGE TO THE CORE MODULE
214

FIG. 5

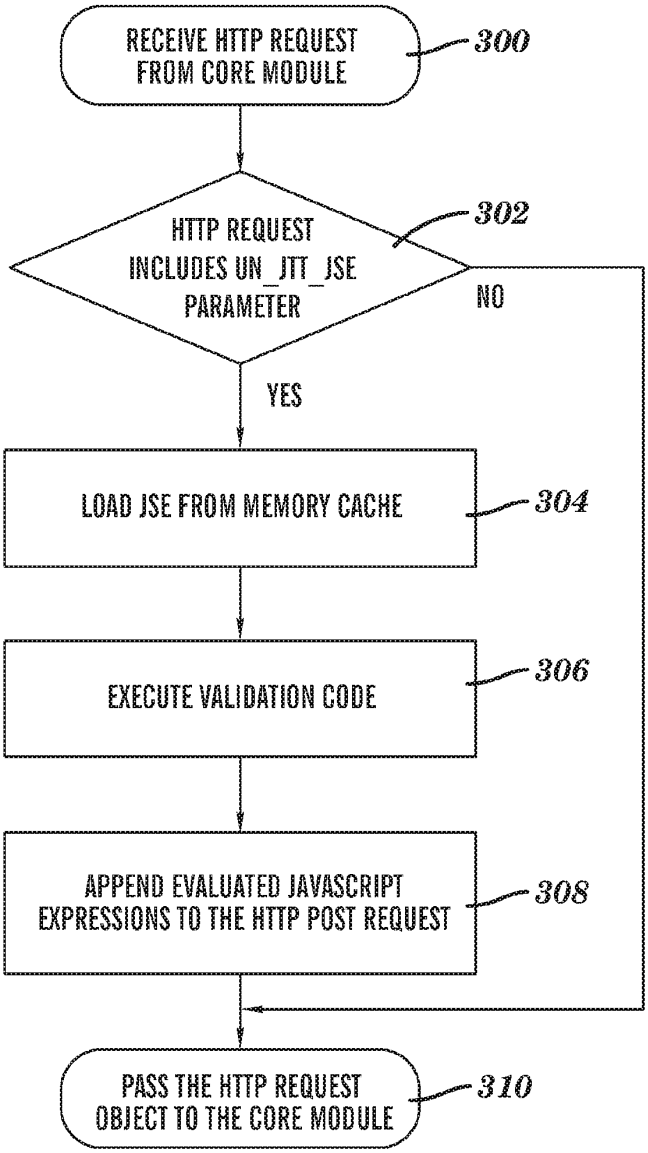


FIG. 6

```
$(document).ready(function() {
        $.get( '/cart.html', function(data) {
                $('#cart').html(data);}
        );
});
```

FIG. 7

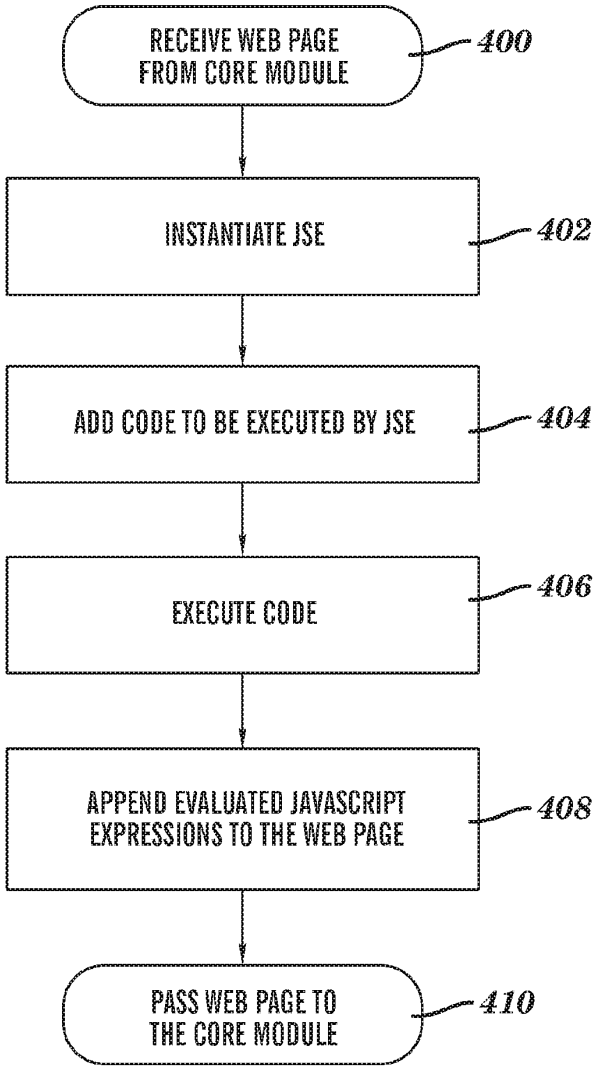

RECEIVE WEB PAGE FROM CORE MODULE
400
INSTANTIATE JSE
402
ADD CODE TO BE EXECUTED BY JSE
404
EXECUTE CODE
406
APPEND EVALUATED JAVASCRIPT EXPRESSIONS TO THE WEB PAGE
408
PASS WEB PAGE TO THE CORE MODULE
410

FIG. 8

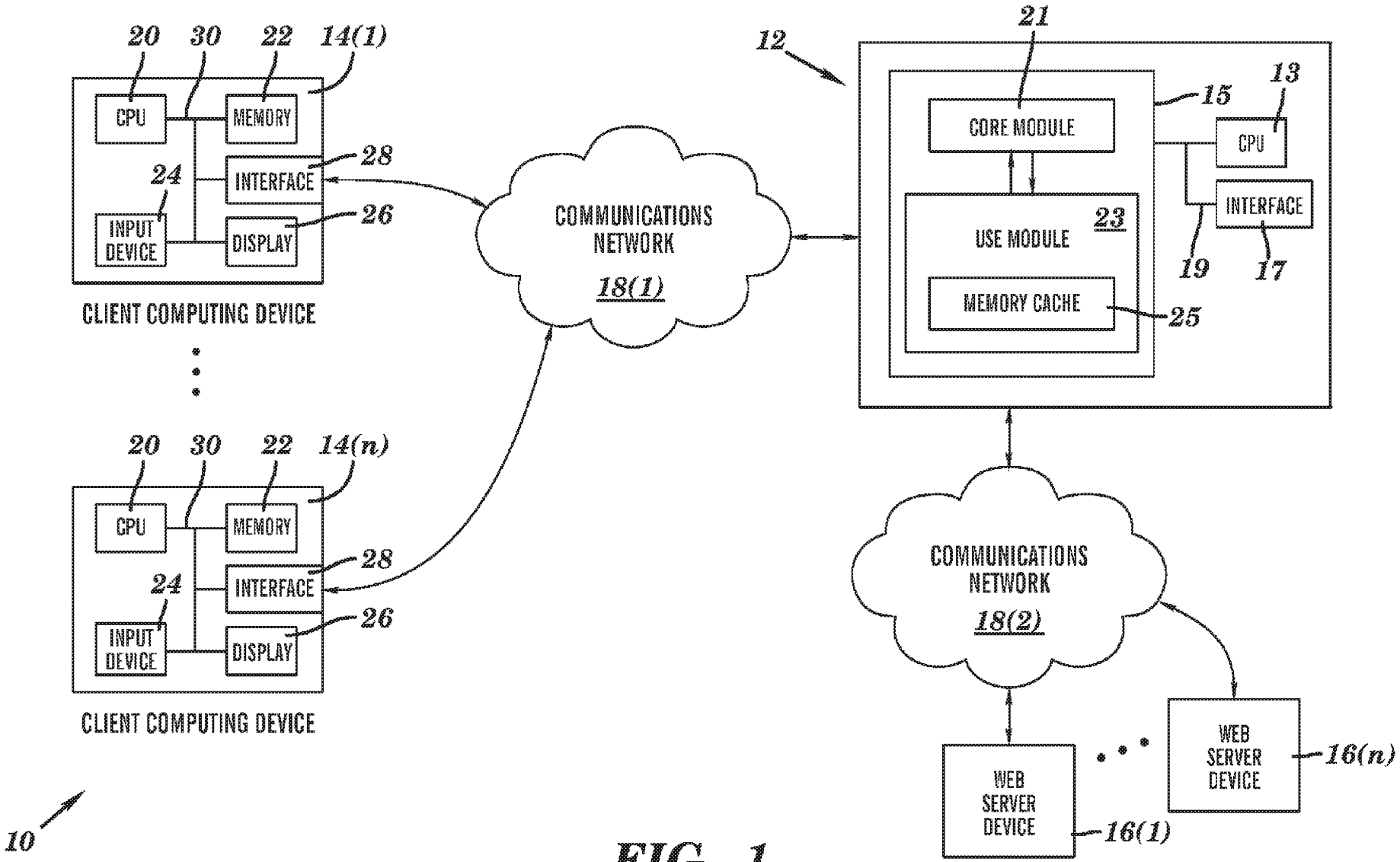


FIG. 1

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2800790 A1 2011/12/15

(21) 2 800 790

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(86) Date de dépôt PCT/PCT Filing Date: 2011/06/10
(87) Date publication PCT/PCT Publication Date: 2011/12/15
(85) Entrée phase nationale/National Entry: 2012/11/23
(86) N° demande PCT/PCT Application No.: US 2011/040021
(87) N° publication PCT/PCT Publication No.: 2011/156739
(30) Priorité/Priority: 2010/06/11 (US12/802,690)

(51) Cl.Int./Int.Cl. *G06F 19/00* (2011.01), *G06Q 10/00* (2012.01), *G06Q 50/00* (2012.01)

(71) Demandeur/Applicant:
USABLENET INC., US

(72) Inventeurs/Inventors:
SCODA, ENRICO, IT;
PEZZANO, SIMONE, IT

(74) Agent: GOWLING LAFLEUR HENDERSON LLP

(54) Titre : APPAREILS DE GESTION DE PROJET LOGICIEL ET LEURS PROCEDES
(54) Title: SOFTWARE PROJECT MANAGEMENT APPARATUSES AND METHODS THEREOF

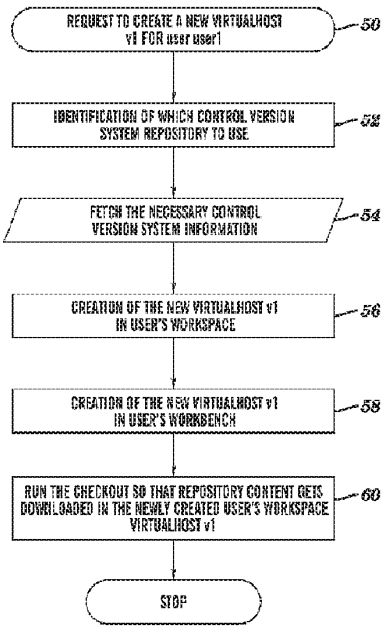


FIG. 4

(57) Abrégé/Abstract:
A method, computer readable medium and apparatus that manages a software project includes assigning one of one or more virtual hosts in one of one or more workspaces in a development computing device to a remote computing device. The development computing device generates at least one link in the one of the one or more workspaces to at least one of one or more working copies of projects in one of one or more work benches in the one of the one or more virtual hosts. The development computing device generates at least one other link in the one of the one or more workspaces to the linked one of the one or more working copies of projects activated in a running area of the development computing device. The development computing device provides access to the activated one of the one or more working copies of projects to the remote computing device to execute one or more tasks.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
15 December 2011 (15.12.2011)

PCT

(10) International Publication Number
WO 2011/156739 A3

(51) International Patent Classification:
G06F 19/00 (2011.01) *G06Q 50/00* (2012.01)
G06Q 10/00 (2012.01)

(21) International Application Number:
PCT/US2011/040021

(22) International Filing Date:
10 June 2011 (10.06.2011)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/802,690 11 June 2010 (11.06.2010) US

(71) Applicant *(for all designated States except US)*: **US-ABLENET INC.** [US/US]; 28 W. 23rd Street, 6th Floor, New York, NY 10010 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **SCODA, Enrico** [IT/IT]; Via Cividina 416/3, I-33035 Matignacco (UD) (IT). **PEZZANO, Simone** [IT/IT]; Via Superiore, 21, I-33100 Udine (IT).

(74) Agents: **LEINBERG, Gunnar, G.** et al.; LECLAIR-RYAN, 290 Linden Oaks, Rochester, NY 14625 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ,

[Continued on next page]

(54) Title: SOFTWARE PROJECT MANAGEMENT APPARATUSES AND METHODS THEREOF

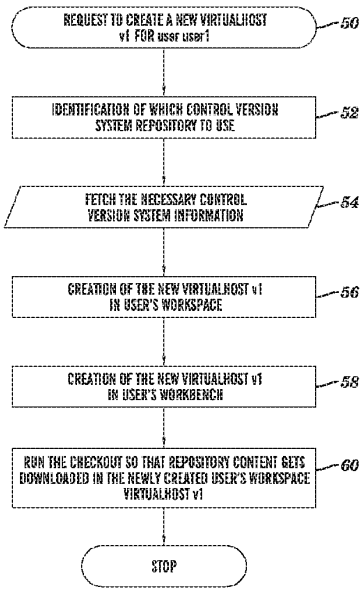


FIG. 4

(57) Abstract: A method, computer readable medium and apparatus that manages a software project includes assigning one of one or more virtual hosts in one of one or more workspaces in a development computing device to a remote computing device. The development computing device generates at least one link in the one of the one or more workspaces to at least one of one or more working copies of projects in one of one or more work benches in the one of the one or more virtual hosts. The development computing device generates at least one other link in the one of the one or more workspaces to the linked one of the one or more working copies of projects activated in a running area of the development computing device. The development computing device provides access to the activated one of the one or more working copies of projects to the remote computing device to execute one or more tasks.

WO 2011/156739 A3

WO 2011/156739 A3

TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(88) Date of publication of the international search report:
12 April 2012

SOFTWARE PROJECT MANAGEMENT APPARATUSES AND METHODS THEREOF

FIELD

[0001] This invention relates to software project management apparatuses and methods and, more particularly, to software project management apparatuses that manage projects which involve execution of tasks at remote workbenches, and methods thereof.

BACKGROUND

[0002] Currently, when managing the development of a software project which utilizes an interpreted computer language, such as HTML XML, JavaScript, JSP, and Python, a working copy of the program will be maintained on a production server. A project manager or managers of the software project will assign projects related to the working copy to developers located at remote working computing stations.

[0003] The developers at each remote working computing station will obtain a remote copy of the program and then develop and test code for the particular assigned project. Typically, each of these remote working computing stations will include an instance of an interpreter installed for execution and testing of the remote copy. When the remote copy is completed by the developer and needs to be published, the remote copy is transferred from the remote working computing station over to the working copy of the program on the production server. At that point, the developer or the project manager verifies the status of the particular assigned project on the production server.

[0004] Unfortunately, when multiple developers are working on different remote copies of the program at the remote working computing stations, there are possible conflicts in concurrency and loss of synchronization among remote copies and working copies. Additionally, each of the remote working computing stations requires a separate instance of the interpreter which can be expensive and also difficult to update and upgrade.

SUMMARY

[0005] A method for managing a software project includes assigning one of one or more virtual hosts in one of one or more workspaces in a development computing device to a remote computing device. The development computing device generates at least one link in the one of the one or more workspaces to at least one of one or more working copies of projects in one of one or more work benches in the one of the one or more virtual hosts. The development computing device generates at least one other link in the one of the one or more workspaces to the linked one of the one or more working copies of projects activated in a running area of the development computing device. The development computing device provides access to the activated one of the one or more working copies of projects to the remote computing device to execute one or more tasks.

[0006] A computer readable medium having stored thereon instructions for managing a software project comprising machine executable code which when executed by at least one processor, causes the processor to perform steps including assigning one of one or more virtual hosts in one of one or more workspaces in a development computing device to a remote computing device. At least one link is generated in the one of the one or more workspaces to at least one of one or more working copies of projects in one of one or more work benches in the one of the one or more virtual hosts. At least one other link is generated in the one of the one or more workspaces to the linked one of the one or more working copies of projects activated in a running area of the development computing device. Access is provided to the activated one of the one or more working copies of projects to the remote computing device to execute one or more tasks.

[0007] A software development management apparatus has one or more processors and a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory including assigning one of one or more virtual hosts in one of one or more workspaces in a development computing device to a remote computing device. At least one link is generated in the one of the one or more workspaces to at least one of one or more working copies of projects in one of one or more work benches in the one of the

one or more virtual hosts. At least one other link is generated in the one of the one or more workspaces to the linked one of the one or more working copies of projects activated in a running area of the development computing device. Access is provided to the activated one of the one or more working copies of projects to the remote computing device to execute one or more tasks.

[0008] This technology provides a number of advantages including providing a more efficient and effective method for managing projects which involve execution of tasks at remote workbenches. This technology enables multiple software developers to accomplish tasks, such as developing, verifying, and testing, on remote computing workbenches and managers to supervise as if they were all located in one location. As a result, the developers at the remote computing workbenches and managers can be located anywhere. Additionally, this technology allows interpreted computer languages source files to be executed on a development server, eliminating the need to install an instance of the interpreter on each remote computing workbench. This helps to reduce costs and also simplifies updating and upgrading of the interpreter. Further, this technology enables managers to gain an easier and more efficient way to manage and control a software development project through its lifecycle.

BRIEF DESCRIPTIONS OF THE DRAWINGS

[0009] FIG. 1 is a partial block and partial functional diagram of an exemplary system environment with a development server;

[0010] FIG. 2 is a functional block diagram of workspaces in the development server shown in FIG. 1;

[0011] FIG. 3 is a functional block diagram of one of a workspace and a corresponding workbench in the development server shown in FIG. 1;

[0012] FIG. 4 is a flow chart of a method for creating a virtual host in the development server;

[0013] FIG. 5 is a flow chart of a method for activating a project in a virtual host in the development server;

[0014] FIG. 6 is a flow chart of a method for managing execution of an activation of a project in a virtual host in the development server;

[0015] FIG. 7 is a flow chart of a method for editing a project in a user workbench in the development server; and

[0016] FIG. 8 is a flow chart of a method for triggering an interpretation of a project in the development server.

DETAILED DESCRIPTION

[0017] An exemplary environment 10 with a development server 12 is illustrated in FIG. 1. The exemplary environment 10 includes the development server 12, a remote workbench computing device 14, a manager computing device 16, and a domain name server 18 are coupled together by one or more communication networks, although other numbers and types of systems, devices, and/or elements in other configurations and environments with other communication network topologies can be used. This technology provides a number of advantages including providing a more efficient and effective method for managing projects which involve execution of tasks at a remote workbench computing devices.

[0018] Referring more specifically to FIG. 1, the development server 12 includes a central processing unit (CPU) or processor, a memory comprising a control version system repository and an interpreter, and an interface system which are coupled together by a bus or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used as well as other types of computing devices can be used. The processor in the development server 12 executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[0019] The memory in the development server 12 stores these programmed instructions for one or more aspects of the present invention as

described and illustrated herein, including the execution of the methods described herein, although some or all of the programmed instructions as well as other data could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor, can be used for the memory in the development server 12. In this example, the control version system repository and the module with programmed instructions for the interpreter are located in the memory of the development server, although each could be in other locations and have other numbers.

[0020] The interface system in the development server 12 is used to operatively couple and communicate between the development server 12 and the remote workbench computing device 14, the manager computing device 16, and the domain name server 18, although other types and numbers of communication networks with other types and numbers of connections and configurations can be used. By way of example only, one or more communication networks can use TCP/IP over Ethernet and industry-standard protocols, including HTTP, HTTPS, WAP, and SOAP, although other types and numbers of communication networks, such as a direct connection, a local area network, a wide area network, modems and phone lines, e-mail, and wireless and hardwire communication technology, each having their own communications protocols, can be used.

[0021] The remote workbench computing device 14 in this example is a computing device that includes a central processing unit (CPU) or processor 20, a memory 22, user input device 24, a display 26, and an interface system 28, and which are coupled together by a bus 30 or other link, although one or more of remote workbench computing device 14 can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 20 in the remote workbench computing device 14 executes a program of stored instructions for one or more aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions. Although one remote workbench computing

device 14 is illustrated in this example, other numbers and types of computing devices could be used.

[0022] The memory 22 in the remote workbench computing device 14 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor 20 can be used for the memory 22 in the remote workbench computing device 14.

[0023] The user input device 24 in the remote workbench computing device 14 is used to input selections, although the user input device could be used to input other types of data and interact with other elements. The user input device can include keypads, touch screens, and/or vocal input processing systems although other types and numbers of user input devices can be used.

[0024] The display 26 in the remote workbench computing device 14 is used to show data and information to the user. The display in the remote workbench computing device 14 is a computer screen display, although other types and numbers of displays could be used.

[0025] The interface system 28 in the remote workbench computing device 14 is used to operatively couple and communicate between the remote workbench computing device 14 and the development server 12, the manager computing device 16, and the domain name server 18 over one or more communication networks, although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[0026] The manager computing device 16 in this example is a computing device that includes a central processing unit (CPU) or processor 32, a memory

34, user input device 36, a display 38, and an interface system 40, and which are coupled together by a bus 42 or other link, although the manager computing device can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 32 in the manager computing device 16 executes a program of stored instructions for one or more aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions. Although one the manager computing device 16 is illustrated in this example, other numbers and types of computing devices could be used

[0027] The memory 34 in the manager computing device 16 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor 32 can be used for the memory 34 in the manager computing device 16.

[0028] The user input device 36 in the manager computing device 16 is used to input selections, although the user input device could be used to input other types of data and interact with other elements. The user input device 36 can include keypads, touch screens, and/or vocal input processing systems although other types and numbers of user input devices can be used.

[0029] The display 38 in the manager computing device 16 is used to show data and information to the user. The display in the manager computing device 16 is a computer screen display, although other types and numbers of displays could be used.

[0030] The interface system 40 in the manager computing device 16 is used to operatively couple and communicate between the manager computing device 16 and the developmental management server 12, the remote workbench

computing device 14, and the domain name server 18 over one or more communication networks, although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[0031] The domain name server 18 includes a central processing unit (CPU) or processor, a memory, and an interface system which are coupled together by a bus or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used as well as other types of computing devices can be used. The processor in the domain name server 18 executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein including assigning domain name addresses, although the processor could execute other numbers and types of programmed instructions.

[0032] The memory in the domain name server 18 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, including the execution of the methods described herein, although some or all of the programmed instructions as well as other data could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor, can be used for the memory in the domain name server 18. In this example, the control version system repository and the module with programmed instructions for the interpreter are located in the memory of the development server, although each could be in other locations and have other numbers.

[0033] The interface system in the domain name server 18 is used to operatively couple and communicate between the domain name server 18 and the development server 12, the remote workbench computing device 14, and the manager computing device 16, although other types and numbers of

communication networks with other types and numbers of connections and configurations can be used.

[0034] Although embodiments of the development server 12, the remote workbench computing device 14, manager computing device 16, and domain name server are described and illustrated herein, each can be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s). Additionally, other numbers of the development server 12, the remote workbench computing device 14, manager computing device 16, and domain name server 18 could be used.

[0035] Furthermore, each of the systems of the embodiments may be conveniently implemented using one or more general purpose computer systems, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[0036] In addition, two or more computing systems or devices can be substituted for any one of the systems in any embodiment of the embodiments. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the embodiments. The embodiments may also be implemented on computer system or systems that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[0037] The embodiments may also be embodied as a computer readable medium having instructions stored thereon for one or more aspects of the present invention as described and illustrated by way of the embodiments herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the embodiments, as described and illustrated herein.

[0038] An exemplary method for creating a virtual host in the development server 12 will now be described with reference to FIGS. 1-4. In step 50, a developer, also referred to as user01 or user1 in this example, at the remote workbench computing device 14 submits a request to create a new virtualhost01 for user01 to the development server 12. In step 52, the development server 12 processes the received request and determines which one of one or more control version system repository is appropriate for one or more projects assigned to the developer (user01 in this example) at the remote workstation computing device 14. In step 54, the development server 12 fetches the one of the control version system repositories determined to be appropriate for the one or more projects assigned to the developer (user01 in this example). In step 56, the development server 12 creates the new virtualhost01 in the workspace for user01 at the remote workstation computing device 14 to remotely access as illustrated in FIGS. 2-3. In step 58, the development server 12 creates the new virtualhost01 in the workbench for the developer (user01 in this example) which is at the development server 12 as illustrated in FIG. 3. In step 60, the developement server 12 runs a checkout so that the content for the one or more projects assigned to the developer (user01 in this example) are downloaded in the newly created virtualhost01 in the workspace for the developer (user01 in this example) at the remote workstation computing device 14 to acess via the internet connection in this example.

[0039] An exemplary method for activating a project in a virtual host in the development server will now be described with reference to FIGS. 1-3 and 5. In step 62, a manager at the manager computing device 16 requests to active project1 in the virtualhost01 for the developer (user01 in this example) in the development server 12 for user01 at the remote workbench computing device 14.

[0040] In step 64, the development server 12 determines whether the virtualhost01 for the developer (user01 in this example) exists. If in step 64 the development server 12 determines the virtualhost01 for the developer (user01 in this example) does not exist, then the No branch is taken to step 66. In step 66, the development server 12 creates the virtualhost01 for user01 using the method described with reference to FIG. 4. If in step 64 the development server 12 determines the virtualhost01 for the developer (user01 in this example) does exist, then the Yes branch is taken to step 68. In step 68, the development server 12 runs a control version system update for one or more of the projects assigned to the virtualhost01 on the workspace for the developer (user01 in this example).

[0041] In step 70, the development server 12 determines whether the project1 the manager at the manager computing device 16 has requested to active exists in the virtualhost01 on the workspace for the developer (user01 in this example), although one or more of the projects can be activated in other manners and from other sources. If in step 70 the development server 12 determines the project1 the manager at the manager computing device 16 has requested to active does not exist in the virtualhost01 on the workspace for the developer (user01 in this example), then the No branch is taken to step 72 resulting in an error message, although other manners for indicating an error can be used. If in step 70 the development server 12 determines the project1 the manager at the manager computing device 16 has requested to active does exist in the virtualhost01 on the workspace for the developer (user01 in this example), then the Yes branch is taken to step 74 where the execution of the activation of a project1 is carried out as explained in greater detail with reference to FIG. 6.

[0042] An exemplary method for managing execution of an activation of a project in a virtual host in the development server 12 will now be described with reference to FIGS. 1-3 and 6. In step 76, the development server 12 begins the execution of the activation of a project1 in the virtualhost01 in the development server 12. The developer at the remote workstation computing device 14 accesses the activated project1 in the virtualhost01 in the development server 12 remotely via an internet connection, although other manners for accessing the project can be used.

[0043] In step 78 the development server 12 determines whether the project1 exists in the workspace for user01 for the developer at the remote workstation computing device 14 to execute one or more tasks. If in step 78 the development server 12 determines the project1 does not exist in the workspace for the developer (user01 in this example), then the No branch is taken to step 80 resulting in an error message sent to the remote workstation computing device 14, although other manners for indicating an error can be used. If in step 78 the development server 12 determines the project1 does exist in the workspace for the developer (user01 in this example), then the Yes branch is taken to step 82.

[0044] In step 82 the development server 12 determines whether the workspace for the developer (user01 in this example) has an associated workbench for user 01 in the virtualhost 01. If in step 82 the development server 12 determines the workspace for user01 does not have an associated workbench for the developer (user01 in this example), then the No branch is taken to step 84. In step 84, the development server 12 creates a workbench for the developer (user01 in this example) associated with the workspace for user01 in the virtualhost 01. If in step 82 the development server 12 determines the workspace for the developer (user01 in this example) does have an associated workspace for user01, then the Yes branch is taken to step 86. In step 86, the development server 12 creates a link between the project1 in the workspace for the developer (user01 in this example) and the project1 in the workbench for the developer (user01 in this example) as shown in FIG. 3. In step 88, the development server 12 creates a link from the project1 in workbench for user01 to a running area in the development server 12 as illustrated in FIG. 1.

[0045] An exemplary method for editing a project in a user workbench in the development server 12 will now be described with reference to FIGS. 1-3 and 7. In step 90, a developer at the remote workstation computing device 12 begins this method for accessing a project, such as a computer file by way of example, for editing or other tasks. In step 92, the remote workstation computing device 12 establishes a secure channel with the development system 12 using a remote file system technology, although other manners for establishing a secure or unsecure connection could be used. The domain name server 18 is used to assign an IP

addresses to the virtualhost in the development server 12 which is used by the remote workstation computing device 14 to access and interact with one or more activated projects, although other manners for connecting and communicating between the devices can be used.

[0046] In step 94 the development server 12 determines whether the developer (user01) at the remote workstation computing device 12 is permitted to access the workbench for user01. If in step 94 the development server 12 determines the developer (user01) at the remote workstation computing device 12 is not permitted to access the workbench for developer (user01), then the No branch is taken to step 96 which returns an error message. If in step 94 the development server 12 determines the developer (user01) at the remote workstation computing device 12 is permitted to access the workbench for developer (user01), then the Yes branch is taken to step 98.

[0047] In step 98, the development system 12 synchronizes the links to the one or more projects in the workbench for user01 to the one or more real files for the projects located in the workbench for user01. In step 100, the development system 12 provides access to the synchronized projects to developer (user01) at the remote workstation computing device 12 through the virtualhost, although other manners for providing access can be used.

[0048] An exemplary method for triggering an interpretation of a project in the development server 12 will now be described with reference to FIGS. 1-3 and 8. In step 102, a developer (user01) at the remote workstation computing device 12 begins this method for triggering an interpretation of a project in the development server 12.

[0049] In step 104 the development server 12 determines whether the developer (user01) requesting an interpretation of the project1 is allowed to establish a connection with the virtualhost1, although other devices and manners for requesting an interpretatin can be used, such as a request from a manager at the manager computng device 16. If in step 104 the development server 12 determines the developer (user01) requesting an interpretation of the project1 is

not allowed to establish a connection with the virtualhost1, then the No branch is taken to step 106 where the request is rejected. If in step 104 the development server 12 determines the developer (user01) requesting an interpretation of the project1 is allowed to establish a connection with the virtualhost1, then the Yes branch is taken to step 108.

[0050] In step 108 the development server 12 determines whether the requested project1 or other file exists in the running area for interpretation. If in step 108 the development server 12 determines the requested project1 does not exist in the running area for interpretation, then the No branch is taken to step 110 where an error message is returned. If in step 108 the development server 12 determines the requested project1 does exist in the running area for interpretation, then the Yes branch is taken to step 112.

[0051] In step 112, the development server 12 synchronizes the project1 in the running area with the project1 in the workspace for user01. In step 114, the development server 12 executes the interpreter to interpret the instructions in project1 or other file being interpreted. In step 116 the development server prints the output generated by the interpretation, although the output could be handled or stored in other manners.

[0052] Accordingly as illustrated and described herein, this technology provides a remote workbench and a remote virtual host. The remote workbench provides a working area in the development server 12 that can be accessed remotely by software developers at a remote workstation computing device and lets them work on tasks on the project as if they were located locally. The remote virtual host is a virtual host that is assigned to the developer through which the developer can launch and execute tasks on any project that is in its workbench. The development server 12 in the examples described herein is TCP/IP accessible, executes the programmed instructions for one or more aspects of the invention and is where the interpreter is installed. Additionally, with this technology, a project manager at a manager computing device 16 can verify with the development server 12 the status of any project at any moment. Even though working remotely, the developers at each remote workbench computing device never face

concurrency because every developer owns a working copy in the development server 12.

[0053] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for managing a software project, the method comprising:

assigning one of one or more virtual hosts in one of one or more workspaces in a development computing device to a remote computing device;

generating with the development computing device at least one link in the one of the one or more workspaces to at least one of one or more working copies of projects in one of one or more work benches in the one of the one or more virtual hosts;

generating with the development computing device at least one other link in the one of the one or more workspaces to the linked one of the one or more working copies of projects activated in a running area of the development computing device; and

providing with the development computing device access to the activated one of the one or more working copies of projects to the remote computing device to execute one or more tasks.

2. The method as set forth in claim 1 further comprising:

linking with the development computing device the activated one of the one or more working copies of projects to one of one or more control version repositories; and

replicating with the development computing device and the one of one or more control version repositories each of the executed one or more tasks on the activated one of the one or more working copies of projects on the other one or more workspaces.

3. The method as set forth in claim 2 further comprising identifying with the development computing device which one of the one of one or more control version repositories to link with the activated one of the one or more working copies of projects.

4. The method as set forth in claim 1 further comprising:

receiving a request at the development computing device to interpret the activated one of the one or more working copies of projects; and

interpreting with an interpreter in the development computing device the activated one of the one or more working copies of projects.

5. The method as set forth in claim 4 further comprising at least one of updating and upgrading the interpreter in the development computing device.

6. The method as set forth in claim 1 further comprising:

receiving at the development computing device an activation request for one of the one or more working copies of projects; and

activating with the development computing device the requested one of the one or more working copies of projects to create the activated one of the one or more working copies of projects.

7. The method as set forth in claim 1 further comprising creating with the development computing device the one or more virtual hosts.

8. The method as set forth in claim 7 further comprising associating a domain name for each of the one or more virtual hosts to an IP address for the development computing device.

9. A computer readable medium having stored thereon instructions for managing a software project comprising machine executable code which when executed by at least one processor, causes the processor to perform steps:

assigning one of one or more virtual hosts in one of one or more workspaces in a development computing device to a remote computing device;

generating at least one link in the one of the one or more

workspaces to at least one of one or more working copies of projects in one of one or more work benches in the one of the one or more virtual hosts;

generating at least one other link in the one of the one or more workspaces to the linked one of the one or more working copies of projects activated in a running area of the development computing device; and

providing access to the activated one of the one or more working copies of projects to the remote computing device to execute one or more tasks.

10. The medium as set forth in claim 9 further comprising:

linking the activated one of the one or more working copies of projects to one of one or more control version repositories; and

replicating each of the executed one or more tasks on the activated one of the one or more working copies of projects on the other one or more workspaces.

11. The medium as set forth in claim 10 further comprising identifying which one of the one of one or more control version repositories to link with the activated one of the one or more working copies of projects.

12. The medium as set forth in claim 9 further comprising:

receiving a request to interpret the activated one of the one or more working copies of projects; and

interpreting with an interpreter the activated one of the one or more working copies of projects.

13. The medium as set forth in claim 12 further comprising at least one of updating and upgrading the interpreter.

14. The medium as set forth in claim 9 further comprising:

receiving an activation request for one of the one or more working copies of projects; and

activating the requested one of the one or more working

copies of projects to create the activated one of the one or more working copies of projects.

15. The medium as set forth in claim 9 further comprising creating the one or more virtual hosts.

16. The medium as set forth in claim 15 further comprising associating a domain name for each of the one or more virtual hosts to an IP address for the development computing device.

17. A software development management apparatus comprising:

one or more processors;

a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory comprising:

assigning one of one or more virtual hosts in one of one or more workspaces in a development computing device to a remote computing device;

generating at least one link in the one of the one or more workspaces to at least one of one or more working copies of projects in one of one or more work benches in the one of the one or more virtual hosts;

generating at least one other link in the one of the one or more workspaces to the linked one of the one or more working copies of projects activated in a running area of the development computing device; and

providing access to the activated one of the one or more working copies of projects to the remote computing device to execute one or more tasks.

18. The apparatus as set forth in claim 17 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising:

linking the activated one of the one or more working copies of projects to one of one or more control version repositories; and

replicating each of the executed one or more tasks on the activated one of the one or more working copies of projects on the other one or more workspaces.

19. The apparatus as set forth in claim 18 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising identifying which one of the one of one or more control version repositories to link with the activated one of the one or more working copies of projects.

20. The apparatus as set forth in claim 17 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising:

receiving a request to interpret the activated one of the one or more working copies of projects; and

interpreting with an interpreter the activated one of the one or more working copies of projects.

21. The apparatus as set forth in claim 20 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising at least one of updating and upgrading the interpreter.

22. The apparatus as set forth in claim 17 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising:

receiving an activation request for one of the one or more working copies of projects; and

activating the requested one of the one or more working copies of projects to create the activated one of the one or more working copies of projects.

23. The apparatus as set forth in claim 17 wherein the one or

more processors is further configured to execute programmed instructions stored in the memory further comprising creating the one or more virtual hosts.

24. The apparatus as set forth in claim 23 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising associating a domain name for each of the one or more virtual hosts to an IP address for the development computing device.

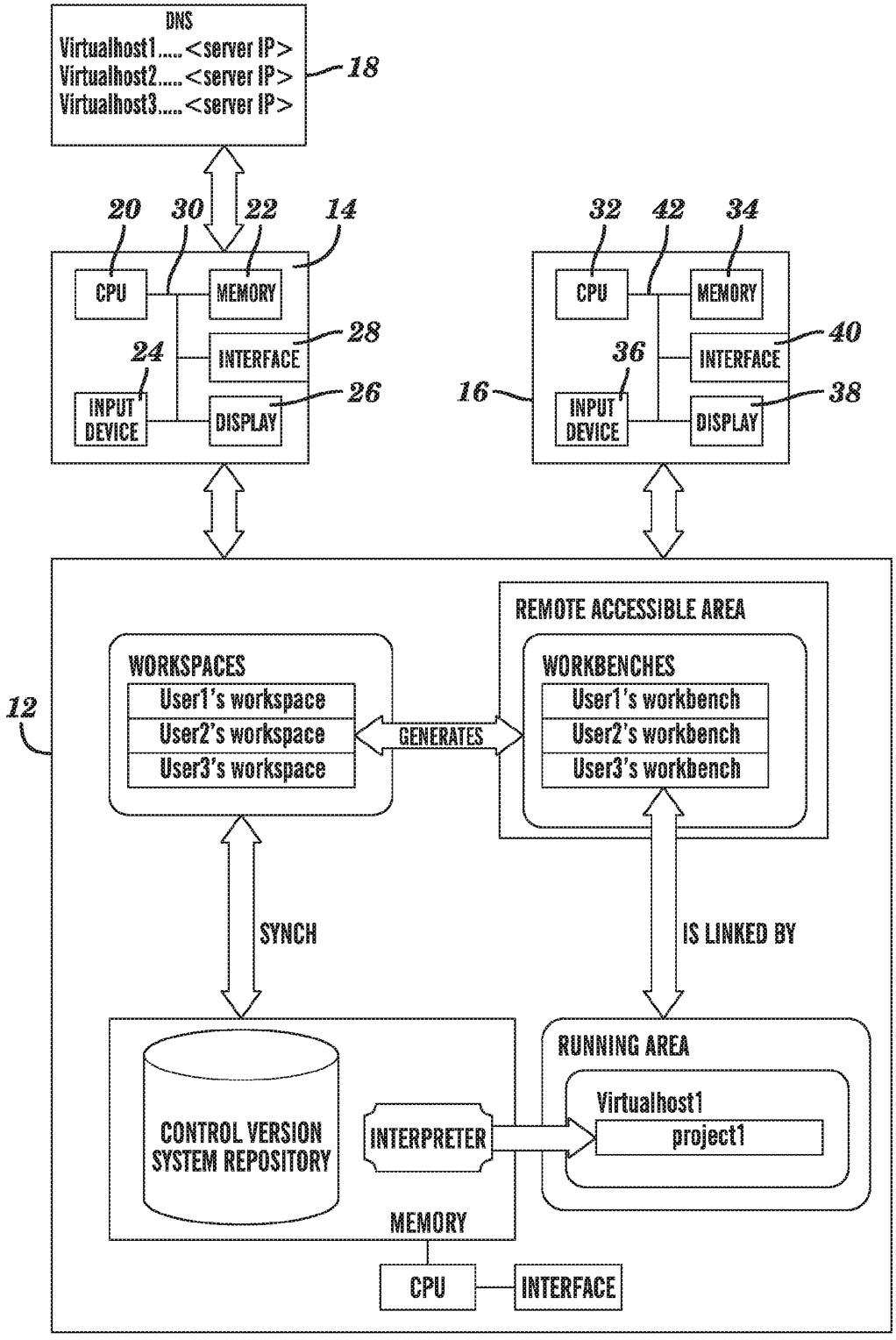

DNS
Virtualhost1.....<server IP>
Virtualhost2.....<server IP>
Virtualhost3.....<server IP>
18
20
30
22
14
CPU
MEMORY
INTERFACE
28
24
INPUT DEVICE
DISPLAY
26
32
42
34
CPU
MEMORY
36
INTERFACE
40
16
INPUT DEVICE
DISPLAY
38
12
REMOTE ACCESSIBLE AREA
WORKSPACES
User1's workspace
User2's workspace
User3's workspace
GENERATES
WORKBENCHES
User1's workbench
User2's workbench
User3's workbench
SYNCH
IS LINKED BY
CONTROL VERSION SYSTEM REPOSITORY
INTERPRETER
RUNNING AREA
Virtualhost1
project1
MEMORY
CPU
INTERFACE

FIG. 1

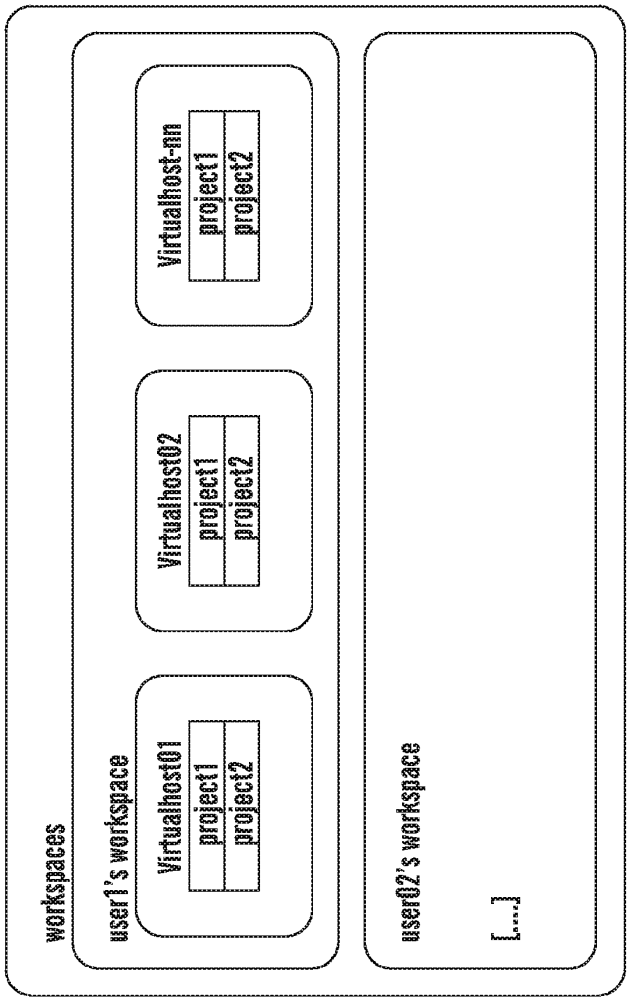

workspaces
user1's workspace
Virtualhost01
project1
project2
Virtualhost02
project1
project2
Virtualhost-nn
project1
project2
user02's workspace
[...]

FIG. 2

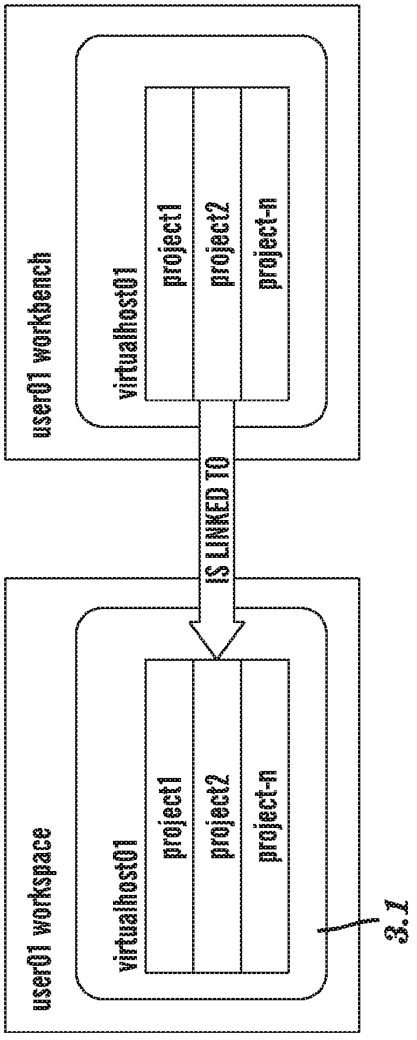

user01 workbench
virtualhost01
project1
project2
project-n
IS LINKED TO
user01 workspace
virtualhost01
project1
project2
project-n
3.1

FIG. 3

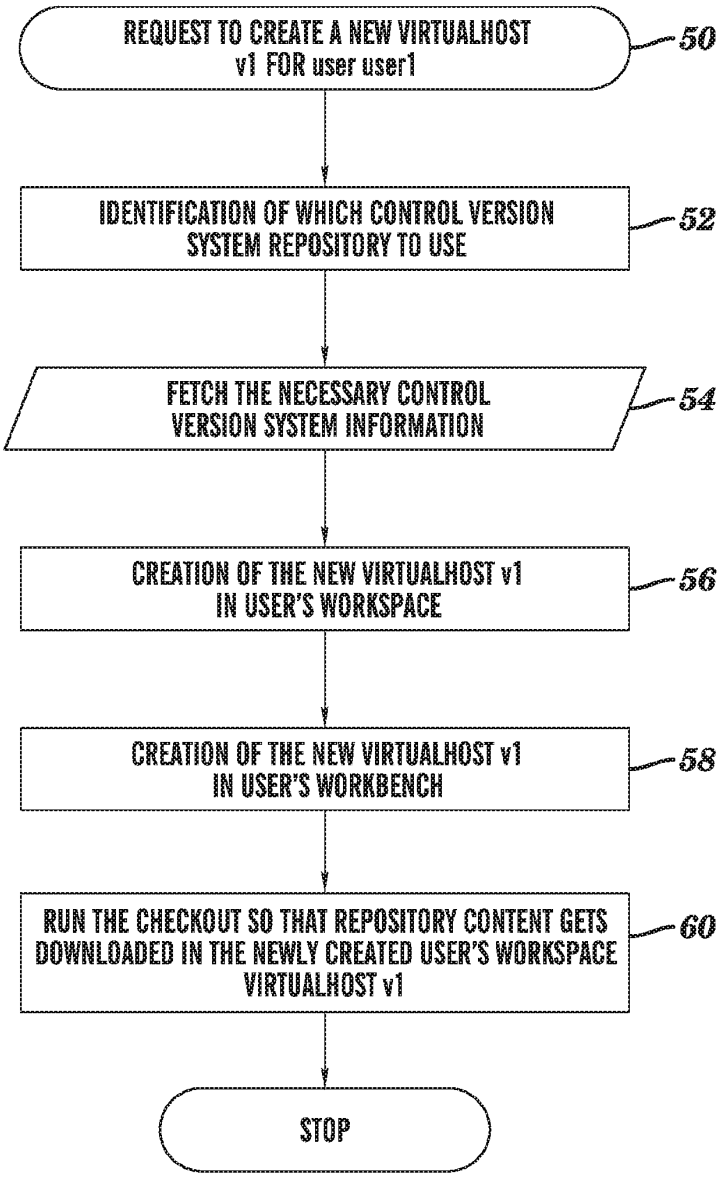

REQUEST TO CREATE A NEW VIRTUALHOST v1 FOR user user1
50
IDENTIFICATION OF WHICH CONTROL VERSION SYSTEM REPOSITORY TO USE
52
FETCH THE NECESSARY CONTROL VERSION SYSTEM INFORMATION
54
CREATION OF THE NEW VIRTUALHOST v1 IN USER'S WORKSPACE
56
CREATION OF THE NEW VIRTUALHOST v1 IN USER'S WORKBENCH
58
RUN THE CHECKOUT SO THAT REPOSITORY CONTENT GETS DOWNLOADED IN THE NEWLY CREATED USER'S WORKSPACE VIRTUALHOST v1
60
STOP

FIG. 4

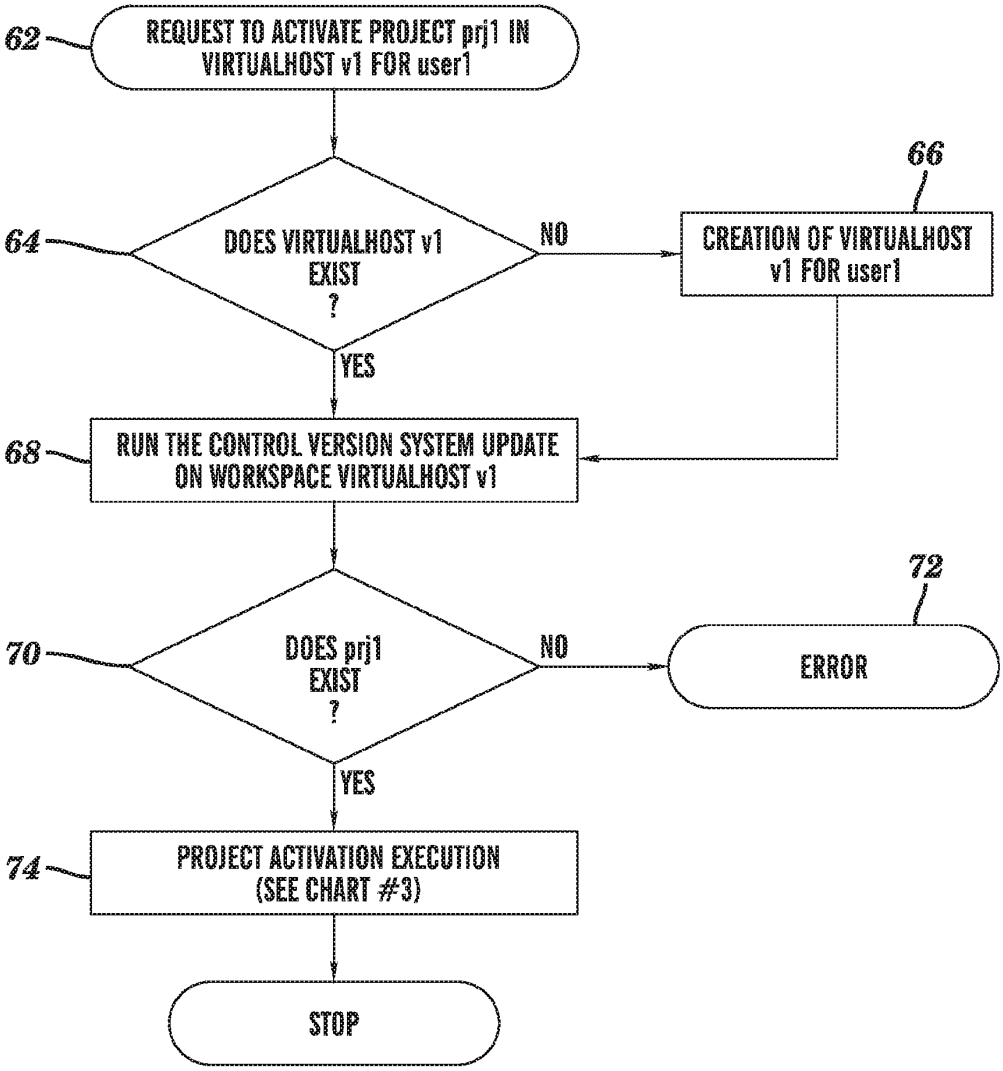

62
REQUEST TO ACTIVATE PROJECT prj1 IN VIRTUALHOST v1 FOR user1
64
DOES VIRTUALHOST v1 EXIST ?
NO
66
CREATION OF VIRTUALHOST v1 FOR user1
YES
68
RUN THE CONTROL VERSION SYSTEM UPDATE ON WORKSPACE VIRTUALHOST v1
70
DOES prj1 EXIST ?
NO
72
ERROR
YES
74
PROJECT ACTIVATION EXECUTION (SEE CHART #3)
STOP

FIG. 5

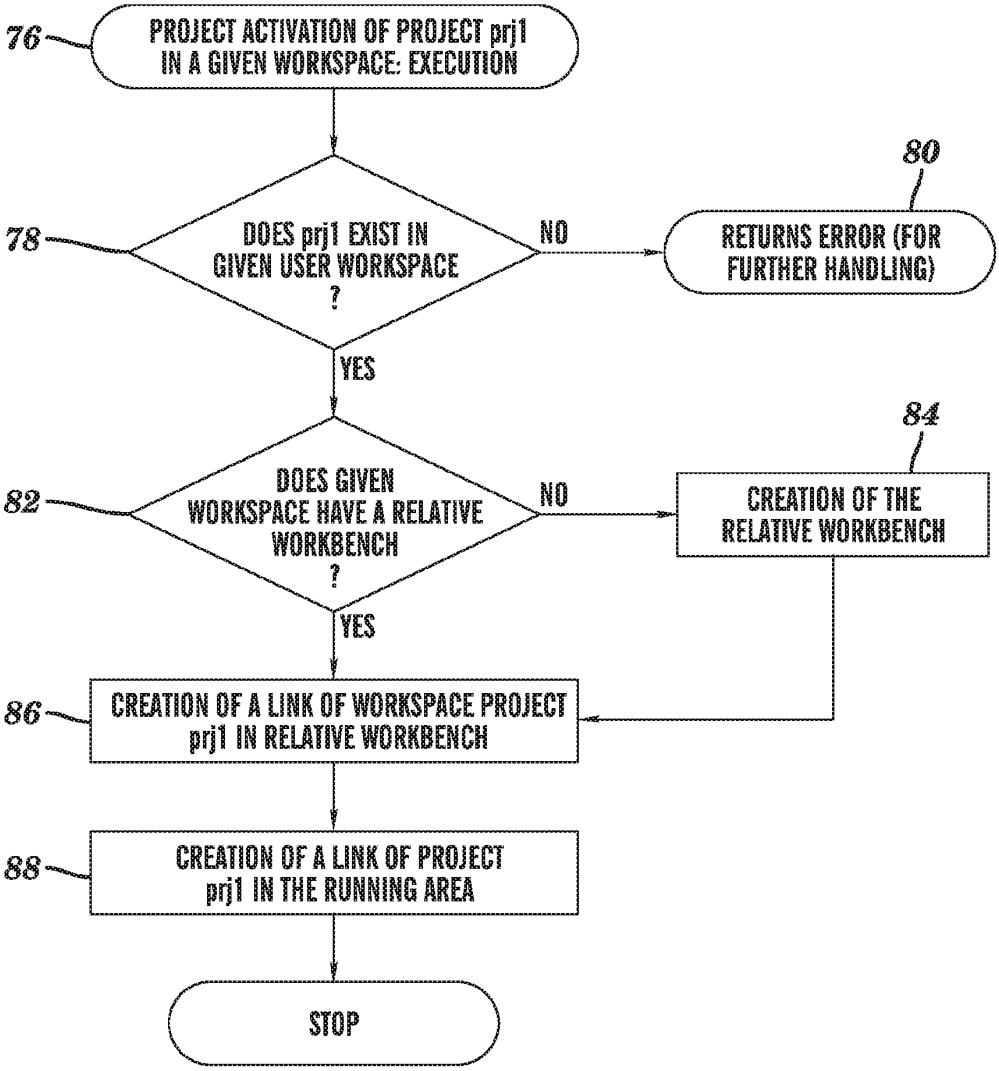

76
PROJECT ACTIVATION OF PROJECT prj1 IN A GIVEN WORKSPACE: EXECUTION
78
DOES prj1 EXIST IN GIVEN USER WORKSPACE ?
NO
80
RETURNS ERROR (FOR FURTHER HANDLING)
YES
82
DOES GIVEN WORKSPACE HAVE A RELATIVE WORKBENCH ?
NO
84
CREATION OF THE RELATIVE WORKBENCH
YES
86
CREATION OF A LINK OF WORKSPACE PROJECT prj1 IN RELATIVE WORKBENCH
88
CREATION OF A LINK OF PROJECT prj1 IN THE RUNNING AREA
STOP

FIG. 6

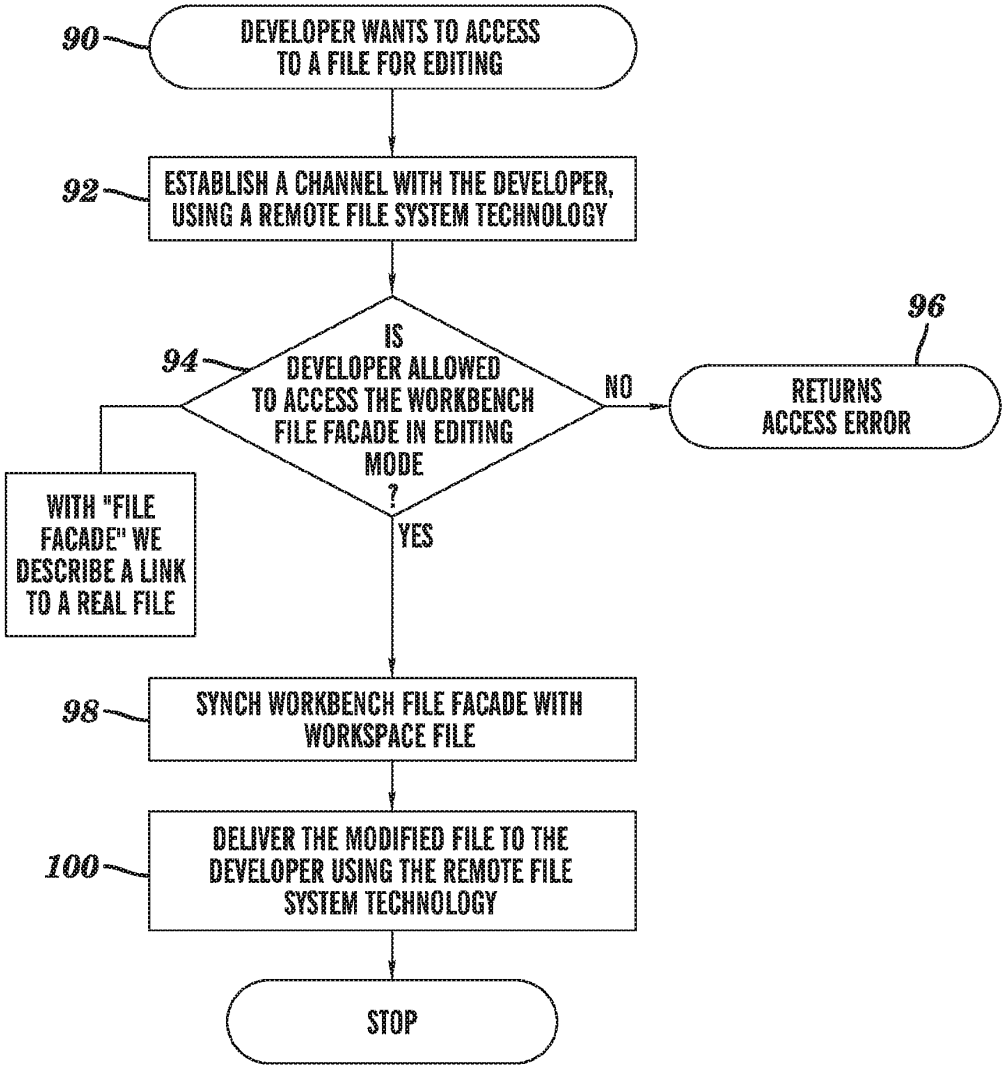

90
DEVELOPER WANTS TO ACCESS TO A FILE FOR EDITING
92
ESTABLISH A CHANNEL WITH THE DEVELOPER, USING A REMOTE FILE SYSTEM TECHNOLOGY
94
IS DEVELOPER ALLOWED TO ACCESS THE WORKBENCH FILE FACADE IN EDITING MODE ?
NO
96
RETURNS ACCESS ERROR
WITH "FILE FACADE" WE DESCRIBE A LINK TO A REAL FILE
YES
98
SYNCH WORKBENCH FILE FACADE WITH WORKSPACE FILE
100
DELIVER THE MODIFIED FILE TO THE DEVELOPER USING THE REMOTE FILE SYSTEM TECHNOLOGY
STOP

FIG. 7

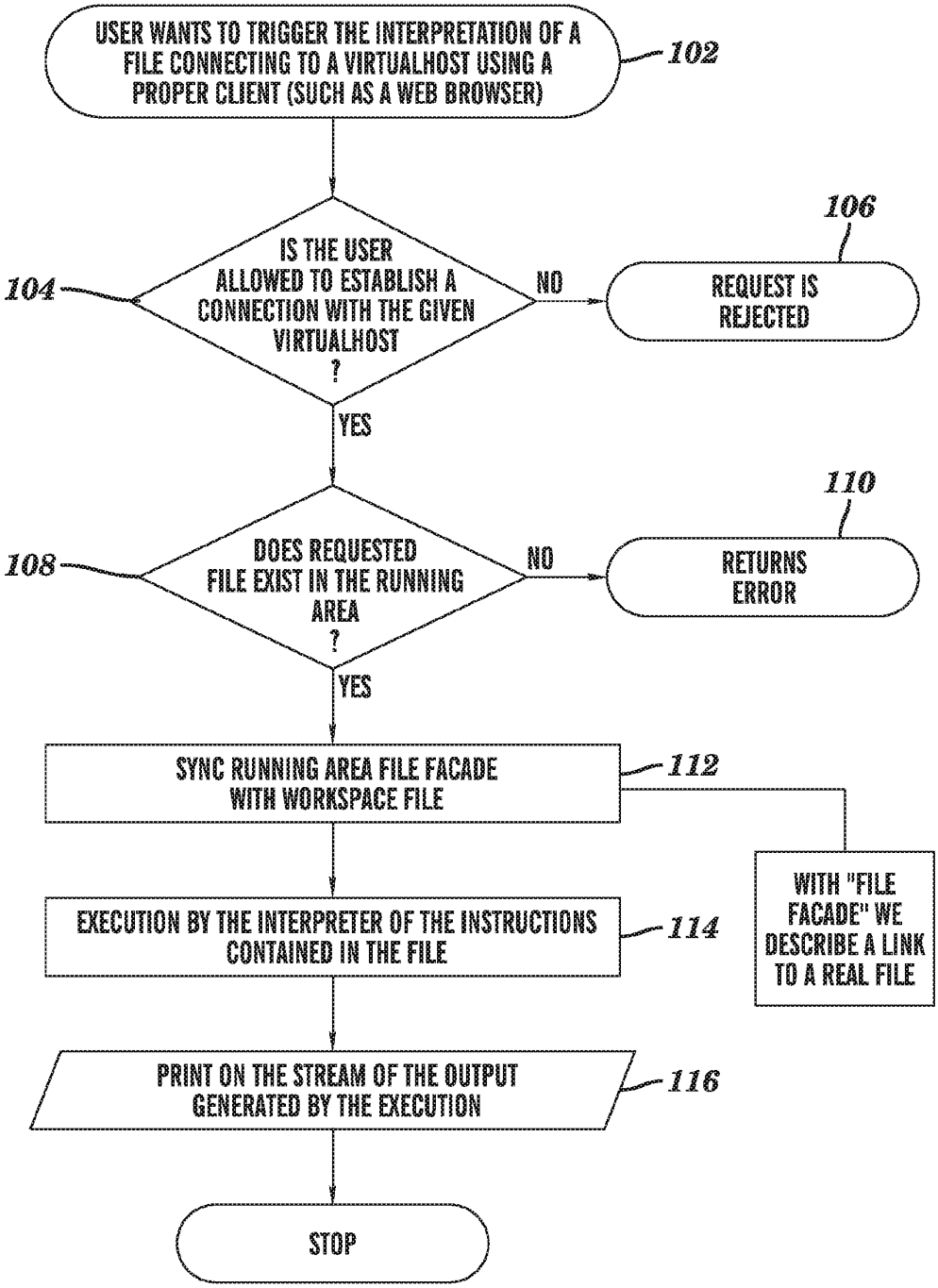

USER WANTS TO TRIGGER THE INTERPRETATION OF A FILE CONNECTING TO A VIRTUALHOST USING A PROPER CLIENT (SUCH AS A WEB BROWSER)
102
IS THE USER ALLOWED TO ESTABLISH A CONNECTION WITH THE GIVEN VIRTUALHOST ?
104
NO
REQUEST IS REJECTED
106
YES
DOES REQUESTED FILE EXIST IN THE RUNNING AREA ?
108
NO
RETURNS ERROR
110
YES
SYNC RUNNING AREA FILE FACADE WITH WORKSPACE FILE
112
WITH "FILE FACADE" WE DESCRIBE A LINK TO A REAL FILE
EXECUTION BY THE INTERPRETER OF THE INSTRUCTIONS CONTAINED IN THE FILE
114
PRINT ON THE STREAM OF THE OUTPUT GENERATED BY THE EXECUTION
116
STOP

FIG. 8

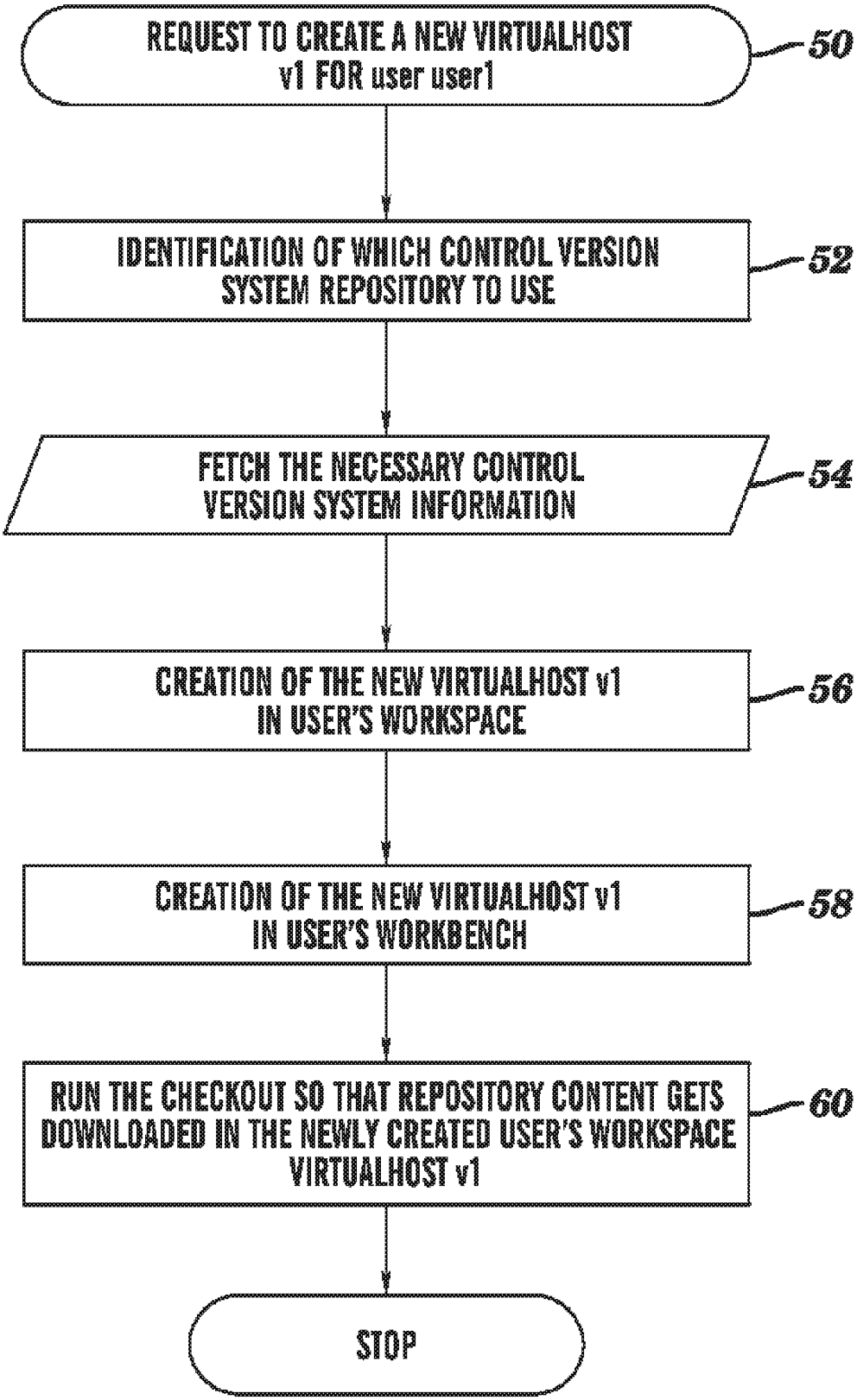

REQUEST TO CREATE A NEW VIRTUALHOST v1 FOR user user1
50
IDENTIFICATION OF WHICH CONTROL VERSION SYSTEM REPOSITORY TO USE
52
FETCH THE NECESSARY CONTROL VERSION SYSTEM INFORMATION
54
CREATION OF THE NEW VIRTUALHOST v1 IN USER'S WORKSPACE
56
CREATION OF THE NEW VIRTUALHOST v1 IN USER'S WORKBENCH
58
RUN THE CHECKOUT SO THAT REPOSITORY CONTENT GETS DOWNLOADED IN THE NEWLY CREATED USER'S WORKSPACE VIRTUALHOST v1
60
STOP

FIG. 4

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2807320 A1 2012/02/23

(21) 2 807 320

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(86) Date de dépôt PCT/PCT Filing Date: 2011/08/17
(87) Date publication PCT/PCT Publication Date: 2012/02/23
(85) Entrée phase nationale/National Entry: 2013/01/31
(86) N° demande PCT/PCT Application No.: US 2011/048058
(87) N° publication PCT/PCT Publication No.: 2012/024380
(30) Priorité/Priority: 2010/08/20 (US12/806,716)

(51) Cl.Int./Int.Cl. *G06F 17/21* (2006.01), *G06F 17/00* (2006.01)
(71) Demandeur/Applicant:
USABLENET, INC., US

(72) Inventeur/Inventor:
SCODA, ENRICO, IT

(74) Agent: GOWLING LAFLEUR HENDERSON LLP

(54) Titre : PROCEDES POUR ADAPTATION SUPPLEMENTAIRE DES TRANSFORMATIONS DE DOCUMENTS XSL EN HTML ET DISPOSITIFS ASSOCIES

(54) Title: METHODS FOR FURTHER ADAPTING XSL TO HTML DOCUMENT TRANSFORMATIONS AND DEVICES THEREOF

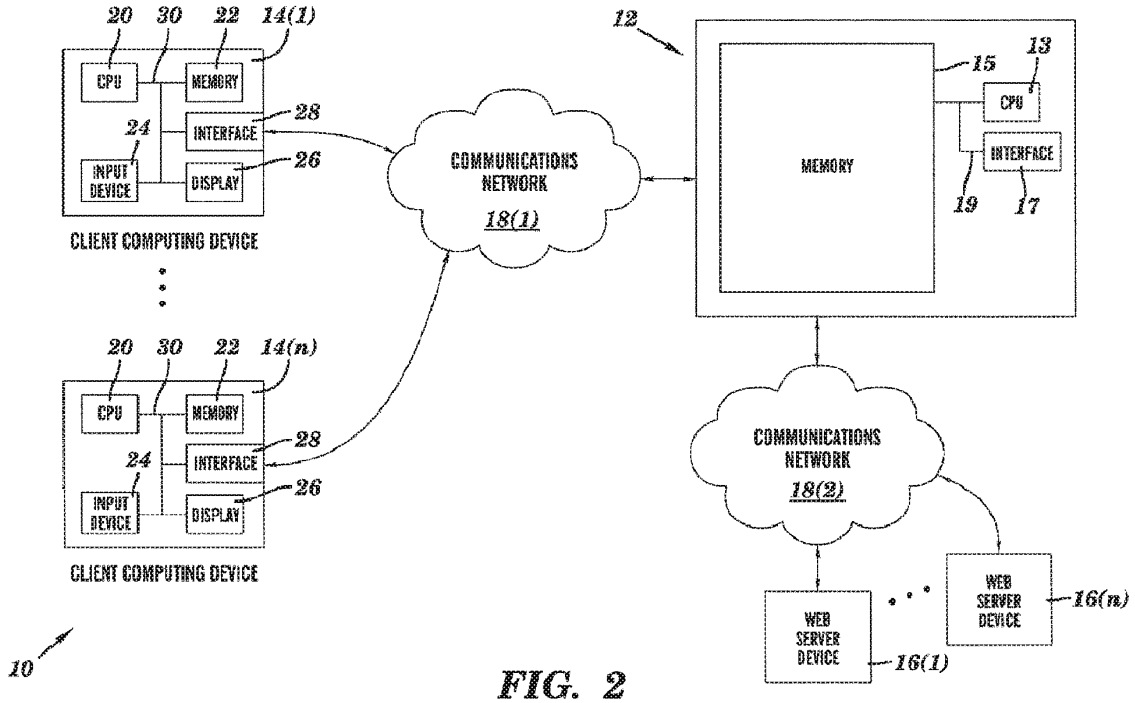


FIG. 2

(57) Abrégé/Abstract:
A method, computer readable medium and apparatus for further adapting XSL to HTML document transformations includes identifying with a web computing device one or more rules in an HTML document. An action associated with each of the identified one or more rules is identified with the web computing device. The identified actions are filtered with the web computing device based on one or more filtering rules when two or more of the identified actions have a match. The remaining identified actions after the filtering are applied with the web computing device to transform the one or more rules in the HTML document. The transformed HTML document is provided by the web computing device.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
23 February 2012 (23.02.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/024380 A3

(51) International Patent Classification:
G06F 17/21 (2006.01) *G06F 17/00* (2006.01)

(21) International Application Number:
PCT/US2011/048058

(22) International Filing Date:
17 August 2011 (17.08.2011)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/806,716 20 August 2010 (20.08.2010) US

(71) Applicant *(for all designated States except US)*: **US-ABLENET, INC.** [US/US]; 28 W. 23rd St., 6th Floor, New York, NY 10010 (US).

(72) Inventor: **SCODA, Enrico**; Via Cividina 416/3, I-33035 Martignacco (UD) (IT).

(74) Agents: **LEINBERG, Gunnar, G.** et al.; LeClairRyan, P.C., 290 Linden Oaks, Rochester, NY 14625 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

--- *with international search report (Art. 21(3))*

(88) Date of publication of the international search report:
14 June 2012

(54) Title: METHODS FOR FURTHER ADAPTING XSL TO HTML DOCUMENT TRANSFORMATIONS AND DEVICES THEREOF

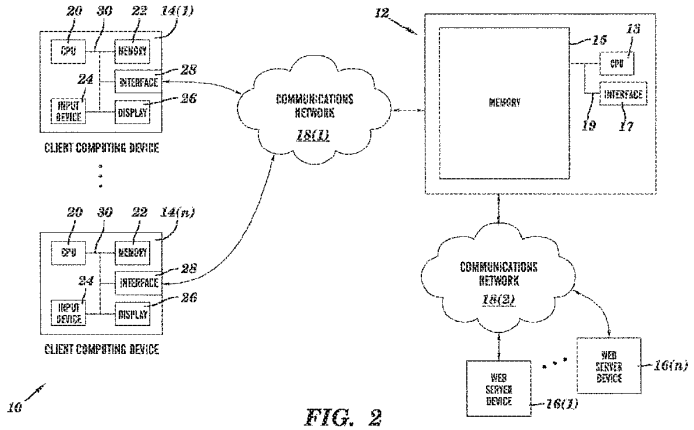


FIG. 2

(57) Abstract: A method, computer readable medium and apparatus for further adapting XSL to HTML document transformations includes identifying with a web computing device one or more rules in an HTML document. An action associated with each of the identified one or more rules is identified with the web computing device. The identified actions are filtered with the web computing device based on one or more filtering rules when two or more of the identified actions have a match. The remaining identified actions after the filtering are applied with the web computing device to transform the one or more rules in the HTML document. The transformed HTML document is provided by the web computing device.

WO 2012/024380 A3

METHODS FOR FURTHER ADAPTING XSL TO HTML DOCUMENT TRANSFORMATIONS AND DEVICES THEREOF

FIELD

[0001] This technology generally relates to methods for adapting eXtensible Stylesheet Language (XSL) to HTML document transformations and devices thereof.

BACKGROUND

[0002] The introduction of eXtensible Markup Language (XML) and the EXtensibleStylesheet Language (XSL) specifications has provided an easy way to transform documents between various formats. This functionality has been included into Web development frameworks, giving them the ability to transform automatically an XML file into a document with different format such as HTML or XHTML, integrating the original data with graphic layout and user interface components. The XSL specifications are based on special constructs called templates that match a single element or a set of similar elements and rewrite them and their content based on instructions defined in the template.

[0003] Unfortunately, a problem arises when the structure of the XML document to process is not well defined. For example, the same element can be used for different purposes inside the XML document and based on these purposes multiple different transformations must be implemented. The problems get even worse when the task involves transforming HTML documents. For example the link element "a" can appear over a thousand times in different sections of a web page, such as in the main navigation bar, in hidden menus, to make images clickable, and as a button to execute JavaScript functions. Writing XSL templates that modify all these elements can increase complexity in an unpredictable way.

[0004] An illustrative example of these difficulties with a simple XSL file managing HTML links ("a" elements) is shown in FIG. 1. As illustrated, the XSL file:

(1) changes the "href" attribute using an XPath extension function called myext:normalize-url();

(2) if the link contains "target" attribute with value "_blank", remove it and set "class" attribute to "external" value, otherwise "class" attribute will get value "internal";

(3) if the "a" content is an image ("img" element), then set new content to image "alt" attribute otherwise apply templates to its children; and

(4) use the "identity" template (last one) to simply copy elements as they are if they are not "a" elements.

[0005] Accordingly, as shown the same instructions have to be written at least twice to keep templates simple and to cover all the combinations of the above transformations. More powerful XSL constructs like name templates or xsl:choose or xsl:if could be utilized and the resulting XSL document will be more optimized, but also will be more complex and less readable.

SUMMARY

[0006] A method for further adapting XSL to HTML document transformations includes identifying with a web computing device one or more rules matching one or more elements in an HTML document. An action associated with each of the identified one or more rules is identified with the web computing device. The identified actions are filtered with the web computing device based on one or more filtering rules when two or more of the identified actions have a match. The remaining identified actions after the filtering are applied with the web computing device to transform the one or more matching elements in the HTML document. The transformed HTML document is provided by the web computing device.

[0007] A computer readable medium having stored thereon instructions processing multiple documents from multiple sites comprising machine executable code which when executed by at least one processor, causes the processor to perform steps including identifying one or more rules matching one

or more elements in an HTML document. An action associated with each of the identified one or more rules is identified. The identified actions are filtered based on one or more filtering rules when two or more of the identified actions have a match. The remaining identified actions after the filtering are applied to transform the one or more matching elements in the HTML document. The transformed HTML document is provided.

[0008] A web computing apparatus includes one or more processors and a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory including identifying one or more rules matching one or more elements in an HTML document. An action associated with each of the identified one or more rules is identified. The identified actions are filtered based on one or more filtering rules when two or more of the identified actions have a match. The remaining identified actions after the filtering are applied to transform the one or more matching elements in the HTML document. The transformed HTML document is provided.

[0009] This technology provides a number of advantages including providing a method, computer readable medium and apparatus that further adapts XSL to HTML document transformations. More specifically, examples of this technology identify a set of similar elements, i.e. sharing same properties, and then defines a set of actions to take on those elements, such as rename, set/change attributes, and set their content. With this technology, if one element is member of two or more sets, then all actions defined for these sets can be applied to the element. This enable smaller sets of instructions to be defined for well defined sets of elements without the need of rewriting same instructions for different sets. These sets of instructions are then transformed into XSL instructions that can be processed by any XSL processor.

BRIEF DESCRIPTION OF THE DRAWINGS

[00010] FIG. 1 is an exemplary XSL file managing HTML links;

[00011] FIG. 2 is a block diagram of an exemplary system with a proxy server configured to adapt XSL to HTML document transformations;

[00012] FIG. 3 is a flow chart of an exemplary method for adapting XSL to HTML document transformations;

[00013] FIG. 4 are three exemplary rules; and

[00014] FIG. 5 is a resulting XSL file from an automatic translation of the rules files shown in FIG. 4 into XSL format.

DETAILED DESCRIPTION

[00015] An exemplary environment 10 with a proxy server 12 configured to further adapt XSL to HTML document transformation is illustrated in FIG. 1, although this technology can be implemented on other types of devices, although this technology can be implemented on other types of devices, such as one of the web server devices 16(1)-16(n) by way of example only. The exemplary environment 10 includes the proxy server or apparatus 12, client devices 14(1)-14(n), the web server devices 16(1)-16(n), and communication networks 18(1)-18(2), although other numbers and types of systems, devices, and/or elements in other configurations and environments with other communication network topologies can be used. This technology provides a number of advantages including providing a method, computer readable medium and an apparatus that further adapts XSL to HTML document transformations.

[00016] Referring more specifically to FIG. 1, the proxy server 12 includes a central processing unit (CPU) or processor 13, a memory 15, and an interface system 17 which are coupled together by a bus 19 or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor 13 in the proxy server 12 executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[00017] The memory 15 in the proxy server 12 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be

stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor 13, can be used for the memory 15 in the proxy server 12.

[00018] The interface system 17 in the proxy server 12 is used to operatively couple and communicate between the proxy server 12 and the client devices 14(1)-14(n) and the web server devices 16(1)-16(n) via the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used. By way of example only, the communication networks 18(1) and 18(2) can use TCP/IP over Ethernet and industry-standard protocols, including HTTP, HTTPS, WAP, and SOAP, although other types and numbers of communication networks, such as a direct connection, a local area network, a wide area network, modems and phone lines, e-mail, and wireless and hardwire communication technology, each having their own communications protocols, can be used.

[00019] Each of the client devices 14(1)-14(n) enables a user to request, get and interact with web pages from one or more web sites hosted by the web server devices 16(1)-16(n) through the proxy server 12 via one or more communication networks, although one or more of the client devices 14(1)-14(n) could access content and utilize other types and numbers of applications from other sources and could provide a wide variety of other functions for the user. Although multiple client devices 14(1)-14(n) are shown, other numbers and types of user computing systems could be used. In this example, the client devices 14(1)-14(n) comprise mobile devices with Internet access that permit a website form page or other retrieved data to be displayed, although each of the client devices 14(1)-14(n). By way of example only, one or more of the client devices 14(1)-14(n) can comprise smart phones, personal digital assistants, or computers.

[00020] Each of client devices 14(1)-14(n) in this example is a computing device that includes a central processing unit (CPU) or processor 20, a memory 22, user input device 24, a display 26, and an interface system 28, and which are coupled together by a bus 30 or other link, although one or more of client devices 14(1)-14(n) can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 20 in each of client devices 14(1)-14(n) executes a program of stored instructions for one or more aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions.

[00021] The memory 22 in each of the client devices 14(1)-14(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor 20 can be used for the memory 22 in each of the client devices 14(1)-14(n.

[00022] The user input device 24 in each of the client devices 14(1)-14(n) is used to input selections, such as requests for a particular website form page or to enter data in fields of a form page, although the user input device could be used to input other types of data and interact with other elements. The user input device can include keypads, touch screens, and/or vocal input processing systems although other types and numbers of user input devices can be used.

[00023] The display 26 in each of the client devices 14(1)-14(n) is used to show data and information to the user, such as website page by way of example only. The display in each of the client devices 14(1)-14(n) is a phone screen display, although other types and numbers of displays could be used depending on the particular type of client device.

[00024] The interface system 28 in each of the client devices 14(1)-14(n) is used to operatively couple and communicate between the client devices 14(1)-14(n) and the proxy server 12 and web server devices 16(1)-16(n) over the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00025] The web server devices 16(1)-16(n) provide one or more pages from one or more web sites for use by one or more of the client devices 14(1)-14(n) via the proxy server 12, although the web server devices 16(1)-16(n) can provide other numbers and types of applications and/or content and can have provide other numbers and types of functions. Although web server devices 16(1)-16(n) are shown for ease of illustration and discussion, other numbers and types of web server systems and devices can be used.

[00026] Each of the web server devices 16(1)-16(n) include a central processing unit (CPU) or processor, a memory, and an interface system which are coupled together by a bus or other link, although each of the web server devices 16(1)-16(n) could have other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor in each of the web server devices 16(1)-16(n) executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[00027] The memory in each of the web server devices 16(1)-16(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated by way of the embodiments, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system

that is coupled to the processor, can be used for the memory in each of the web server devices 16(1)-16(n).

[00028] The interface system in each of the web server devices 16(1)-16(n) is used to operatively couple and communicate between the web server devices 16(1)-16(n) and the proxy server 12 and the client devices 14(1)-14(n) via communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00029] Although embodiments of the proxy server 12, the client devices 14(1)-14(n), and the web server devices 16(1)-16(n), are described and illustrated herein, each of the client devices 14(1)-14(n), the proxy server 12, and the web server devices 16(1)-16(n), can be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s).

[00030] Furthermore, each of the systems of the embodiments may be conveniently implemented using one or more general purpose computer systems, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[00031] In addition, two or more computing systems or devices can be substituted for any one of the systems in any embodiment of the embodiments. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the embodiments. The embodiments may also be implemented on computer system or systems that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless

communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[00032] The embodiments may also be embodied as a computer readable medium having instructions stored thereon for one or more aspects of the present invention as described and illustrated by way of the embodiments herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the embodiments, as described and illustrated herein.

[00033] An exemplary method for further adapting XSL to HTML document transformations with proxy server 12 will now be described with reference to FIGS. 2-5, although again this technology can be executed by other types of devices, such as by one of the web server devices 16(1)-16(n) and without a proxy server by way of example only. Referring more specifically to FIG. 3, in step 100 the proxy server 12 receives an HTML document to transform from one of the web server devices 16(1)-16(n) for one of the client computing devices 14(1)-14(n), although this exemplary method can be executed by other types and numbers of devices. The proxy server 12 traverses the HTML document to identify each element and generate rules for each identified element. In this particular example, the rules files illustrated in FIG. 4 are rules written for HTML element "a", although the HTML document can have other types of and numbers of elements and rules.

[00034] In step 102, the proxy server 12 identifies the action(s) defined for each rule in the rules files shown in FIG. 4 matching the HTML element being analyzed, although other manners for finding the action(s) could be used. The action(s) are statements that when executed change some properties of the matched element and are defined by the XML element(s) that are a child or children of the rule execute section. The match section of the rule is an XPath expression used to identify the other HTML elements matching the rule, although other matching expressions could be used. In this exemplary embodiment the

available actions are: (1) remove-element: to remove the matched element; (2) replace-element: to replace the matched element with a new element (changing its name); (3) linearize-table: to take some (or all) the cells of the matched HTML table and place their content in a different order thanks to multiple instructions called show-cell (for example <show-cell row="3" col="2"/>; (4) set-meta-category: to set the value of a special attribute called un-meta for the matched element; (5)move-bottom: take matched element and move it to the bottom of the document; (6) set-attribute: to set/change the value of a given attribute of matched element; (7) remove-attribute: to remove a given attribute of matched element; (8) set-content: to set new content for the matched element; (9) append-content: to append new content after the last child of current element. It's evident that some elements to be transformed can match two or even more sets of actions. In this case all sets actions will be "eligible" to be applied to the element. The priority/conflict rules defined for the language will be applied to filter actions and to decide which ones to use.

[00035] In step 104, the proxy server 12 determines whether any of the identified actions for the rules match. If in step 104, the proxy server 12 determines there are no matching actions, then the No branch is taken to step 106. In step 106, the proxy server 12 executes the actions on the rules to transform the HTML document. If in step 104, the proxy server 12 determines there are matching actions, then the Yes branch is taken to step 108.

[00036] In step 108, the proxy server 12 applies one or more filtering rules to filter the out the matching actions which are not applicable, although other manners for filtering the matching actions can be used. In this example, the filtering rules are: (1) group all matching actions based on document order of appearance (2) if the action is remove-element, then remove all of the following: remove-element; replace-element; linearize-table; set-meta-category; move-bottom; set-attribute; remove-attribute; set-content; and append-content; (3) if the action is replace-element, then remove all of the following: remove-element; replace-element; and linearize-table; (4) if the action is linearize-table, then remove all of the following: remove-element; replace-element; linearize-table; set-meta-category; set-attribute; remove-attribute; set-content; append-content.

Furthermore remove all preceding: set-meta-category; set-attribute; remove-attribute; set-content; and append-content. (5) if the action is set-meta-category, then remove all following: remove-element, set-meta-category; (6) if action is move-bottom, remove all following: remove-element and move-bottom; (7) if the action is set-attribute, then remove all following: remove-element; set-attribute if name parameter of following action is equal to the name parameter of the matching action; and remove-attribute if name parameter of following action is equal to the name parameter of the matching action; (8) if the action is remove-attribute, then remove all following: remove-element; set-attribute if name parameter of following action is equal to the name parameter of the matching action and remove-attribute if name parameter of following action is equal to the name parameter of the matching action; (9) if the action is set-content, then remove all following: remove-element; set-content and append-content; and (10) if the action is append-content, then remove all following: remove-element; set-content; and append-content.

[00037] In step 110, the proxy server 12 applies the remaining action(s) which remain after the filtering to transform the elements of the HTML document. In step 112, the proxy server 12 provides the transformed elements of the HTML document.

[00038] In this particular example, the XSL file resulting from the automatic translation of the rules files shown in FIG. 4 into the XSL format is illustrated in FIG. 5. As shown, the resulting XSL file can be even more complex than the original XSL file shown in FIG. 1. This is expected since the new language has been created to transfer complexity at the machine level.

[00039] Accordingly, as illustrated and described herein this technology provides a number of advantages including providing a method, computer readable medium and an apparatus that further adapts XSL to HTML document transformations. More specifically, examples of this technology identify a set of similar elements, i.e. sharing the same properties, and then defines a set of actions to take on those elements, such as rename, set/change attributes, and set their content. With this technology, if one element is member of two or more sets, then

all actions defined for these sets can be applied to the element. This enable smaller sets of instructions to be defined for well defined sets of elements without the need of rewriting same instructions for different sets. These sets of instructions are then transformed into XSL instructions that can be processed by any XSL processor.

[00040] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for further adapting XSL to HTML document transformations, the method comprising:

identifying with a web computing device one or more rules matching one or more elements in an HTML document;

identifying with the web computing device an action associated with each of the identified one or more rules;

filtering with the web computing device the identified actions based on one or more filtering rules when two or more of the identified actions have a match; and

applying with the web computing device the remaining identified actions after the filtering to transform the one or more matching elements in the HTML document

providing with the web computing device the transformed HTML document.

2. The method as set forth in claim 1 further comprising determining with the web computing device when two or more of the identified actions have a match.

3. The method as set forth in claim 1 wherein the filtering further comprises:

grouping with the web computing device each of the identified actions which have a match together; and

removing with the web computing device one or more of the actions in each of the groups based on the one or more filtering rules.

4. The method as set forth in claim 3 wherein the one or more filtering rules comprises removing remove-element, replace-element, linearize-table, set-meta-category, move-bottom, set-attribute, remove-attribute, set-content, and append-content when the identified action in the group is remove-element.

5. The method as set forth in claim 3 wherein the one or more filtering rules comprises removing remove-element, replace-element, and linearize-table when the identified action in the group is replace-element.

6. The method as set forth in claim 3 wherein the one or more filtering rules comprises removing remove-element, replace-element, linearize-table, set-meta-category, set-attribute, remove-attribute, set-content, append-content and removing all preceding set-meta-category, set-attribute, remove-attribute, set-content, and append-content when the identified action in the group is linearize-table.

7. The method as set forth in claim 3 wherein the one or more filtering rules comprises removing remove-element and set-meta-category when the identified action in the group is set-meta-category.

8. The method as set forth in claim 3 wherein the one or more filtering rules comprises removing remove-element and move-bottom when the identified action in the group is move-bottom.

9. The method as set forth in claim 3 wherein the one or more filtering rules comprises removing remove-element, set-attribute if name parameter of following action is equal to the name parameter of the matching action, and remove-attribute if name parameter of following action is equal to the name parameter of the matching action when the identified action in the group is set-attribute.

10. The method as set forth in claim 3 wherein the one or more filtering rules comprises removing remove-element, set-attribute if name parameter of following action is equal to the name parameter of the matching action and remove-attribute if name parameter of following action is equal to the name parameter of the matching action when the identified action in the group is remove-attribute.

11. The method as set forth in claim 3 wherein the one or more filtering rules comprises removing remove-element, set-content, and append-content when the identified action in the group is set-content.

12. The method as set forth in claim 3 wherein the one or more filtering rules comprises removing remove-element, set-content, and append-content when the identified action in the group is append-content.

13. The method as set forth in claim 1 wherein one or more of the actions can comprise one or more of remove-element replace-element, linearize-table, set-meta-category, move-bottom, set-attribute, remove-attribute, set-content, and append-content.

14. A computer readable medium having stored thereon instructions processing multiple documents from multiple sites comprising machine executable code which when executed by at least one processor, causes the processor to perform steps comprising:

identifying one or more rules matching one or more elements in an HTML document;

identifying an action associated with each of the identified one or more rules;

filtering the identified actions based on one or more filtering rules when two or more of the identified actions have a match; and

applying the remaining identified actions after the filtering to transform the one or more matching elements in the HTML document

providing the transformed HTML document.

15. The medium as set forth in claim 14 further comprising determining when two or more of the identified actions have a match.

16. The medium as set forth in claim 14 wherein the filtering further comprises:

grouping each of the identified actions which have a match together; and

removing one or more of the actions in each of the groups based on the one or more filtering rules.

17. The medium as set forth in claim 16 wherein the one or more filtering rules comprises removing remove-element, replace-element, linearize-table, set-meta-category, move-bottom, set-attribute, remove-attribute, set-content, and append-content when the identified action in the group is remove-element.

18. The medium as set forth in claim 16 wherein the one or more filtering rules comprises removing remove-element, replace-element, and linearize-table when the identified action in the group is replace-element.

19. The medium as set forth in claim 16 wherein the one or more filtering rules comprises removing remove-element, replace-element, linearize-table, set-meta-category, set-attribute, remove-attribute, set-content, append-content and removing all preceding set-meta-category, set-attribute, remove-attribute, set-content, and append-content when the identified action in the group is linearize-table.

20. The medium as set forth in claim 16 wherein the one or more filtering rules comprises removing remove-element and set-meta-category when the identified action in the group is set-meta-category.

21. The medium as set forth in claim 16 wherein the one or more filtering rules comprises removing remove-element and move-bottom when the identified action in the group is move-bottom.

22. The medium as set forth in claim 16 wherein the one or more filtering rules comprises removing remove-element, set-attribute if name parameter of following action is equal to the name parameter of the matching action, and remove-attribute if name parameter of following action is equal to the name parameter of the matching action when the identified action in the group is set-attribute.

23. The medium as set forth in claim 16 wherein the one or more filtering rules comprises removing remove-element, set-attribute if name parameter of following action is equal to the name parameter of the matching action and remove-attribute if name parameter of following action is equal to the name parameter of the matching action when the identified action in the group is remove-attribute.

24. The medium as set forth in claim 16 wherein the one or more filtering rules comprises removing remove-element, set-content, and append-content when the identified action in the group is set-content.

25. The medium as set forth in claim 16 wherein the one or more filtering rules comprises remvoing remove-element, set-content, and append-content when the identified action in the group is append-content.

26. The medium as set forth in claim 14 wherein one or more of the actions can comprise one or more of remove-element replace-element, linearize-table, set-meta-category, move-bottom, set-attribute, remove-attribute, set-content, and append-content.

27. A web proxy apparatus comprising:

one or more processors;

a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory comprising:

identifying one or more rules matching one or more elements in an HTML document;

identifying an action associated with each of the identified one or more rules;

filtering the identified actions based on one or more filtering rules when two or more of the identified actions have a match; and

applying the remaining identified actions after the filtering to transform the one or more matching elements in the HTML document

providing the transformed HTML document.

28. The apparatus as set forth in claim 27 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising determining when two or more of the identified actions have a match.

29. The apparatus as set forth in claim 27 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for the filtering further comprising:

grouping each of the identified actions which have a match together; and

removing one or more of the actions in each of the groups based on the one or more filtering rules.

30. The apparatus as set forth in claim 29 wherein the one or more filtering rules comprises removing remove-element, replace-element, linearize-table, set-meta-category, move-bottom, set-attribute, remove-attribute, set-content, and append-content when the identified action in the group is remove-element.

31. The apparatus as set forth in claim 30 wherein the one or more filtering rules comprises removing remove-element, replace-element, and linearize-table when the identified action in the group is replace-element.

32. The apparatus as set forth in claim 30 wherein the one or more filtering rules comprises removing remove-element, replace-element,

linearize-table, set-meta-category, set-attribute, remove-attribute, set-content, append-content and removing all preceding set-meta-category, set-attribute, remove-attribute, set-content, and append-content when the identified action in the group is linearize-table.

33. The apparatus as set forth in claim 30 wherein the one or more filtering rules comprises removing remove-element and set-meta-category when the identified action in the group is set-meta-category.

34. The apparatus as set forth in claim 30 wherein the one or more filtering rules comprises removing remove-element and move-bottom when the identified action in the group is move-bottom.

35. The apparatus as set forth in claim 30 wherein the one or more filtering rules comprises removing remove-element, set-attribute if name parameter of following action is equal to the name parameter of the matching action, and remove-attribute if name parameter of following action is equal to the name parameter of the matching action when the identified action in the group is set-attribute.

36. The apparatus as set forth in claim 30 wherein the one or more filtering rules comprises removing remove-element, set-attribute if name parameter of following action is equal to the name parameter of the matching action and remove-attribute if name parameter of following action is equal to the name parameter of the matching action when the identified action in the group is remove-attribute.

37. The apparatus as set forth in claim 30 wherein the one or more filtering rules comprises removing remove-element, set-content, and append-content when the identified action in the group is set-content.

38. The apparatus as set forth in claim 30 wherein the one or more filtering rules comprises remvoing remove-element, set-content, and append-content when the identified action in the group is append-content.

39. The apparatus as set forth in claim 27 wherein one or more of the actions can comprise one or more of remove-element replace-element, linearize-table, set-meta-category, move-bottom, set-attribute, remove-attribute, set-content, and append-content.

```
<?xml version="1.0" encoding= "UTF-8"?>
<xsl:stylesheet xnlns:xsl-"http://www.w3.org/1999/XSL/Transform" version="1.0")

    <xsl:template match="a[@target='_blank' and img]">
        <a href="{myext:normalize-url(@href)}" class="external">,
            <xsl:value-of select="img/@alt"/>
        </a>
    </xsl:template>

    <xsl:template match="a[@target='_blank']">
        <a href="{myext:normalize-url(@href)}" class="external">,
            <xsl:apply-templates/>
        </a>
    </xsl:template>

    <xsl:template match="a[img]">
        <a href="{myext:normalize-url(@href)}" class="internal">,
            <xst:value-of select="img/@alt"/>
        </a>
    </xsl:template>

    <xsl:template match="a">
        <a hret="{myext:normalize-url(@href)}" class=internal">,
            <xsl:apply-templates/>
        </a>
    </xsl:template>

    <xsl:template match="*">
        <xsl:copy>
            <xsl:copy-of select="@*"/>
            </xsl:apply-templates/>
        </xsl:copy>
    </xsl:template>

</xsl:stylesheet>
```

FIG. 1

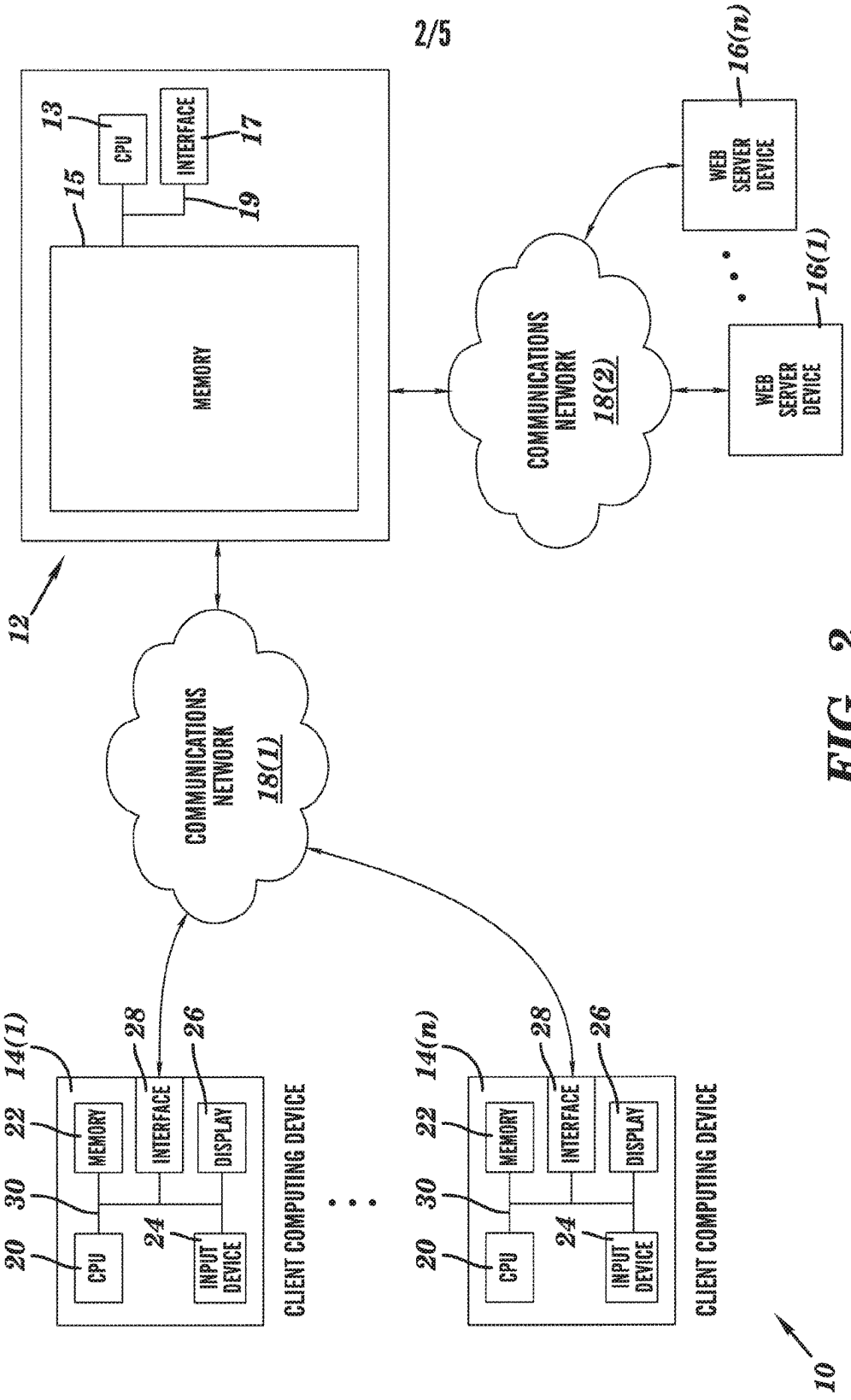

12
13
CPU
INTERFACE
17
15
19
MEMORY
COMMUNICATIONS NETWORK 18(2)
16(n)
WEB SERVER DEVICE
16(1)
WEB SERVER DEVICE
COMMUNICATIONS NETWORK 18(1)
14(1)
28
26
22
MEMORY
INTERFACE
DISPLAY
30
20
CPU
24
INPUT DEVICE
CLIENT COMPUTING DEVICE
14(n)
28
26
22
MEMORY
INTERFACE
DISPLAY
30
20
CPU
24
INPUT DEVICE
CLIENT COMPUTING DEVICE
10

FIG. 2

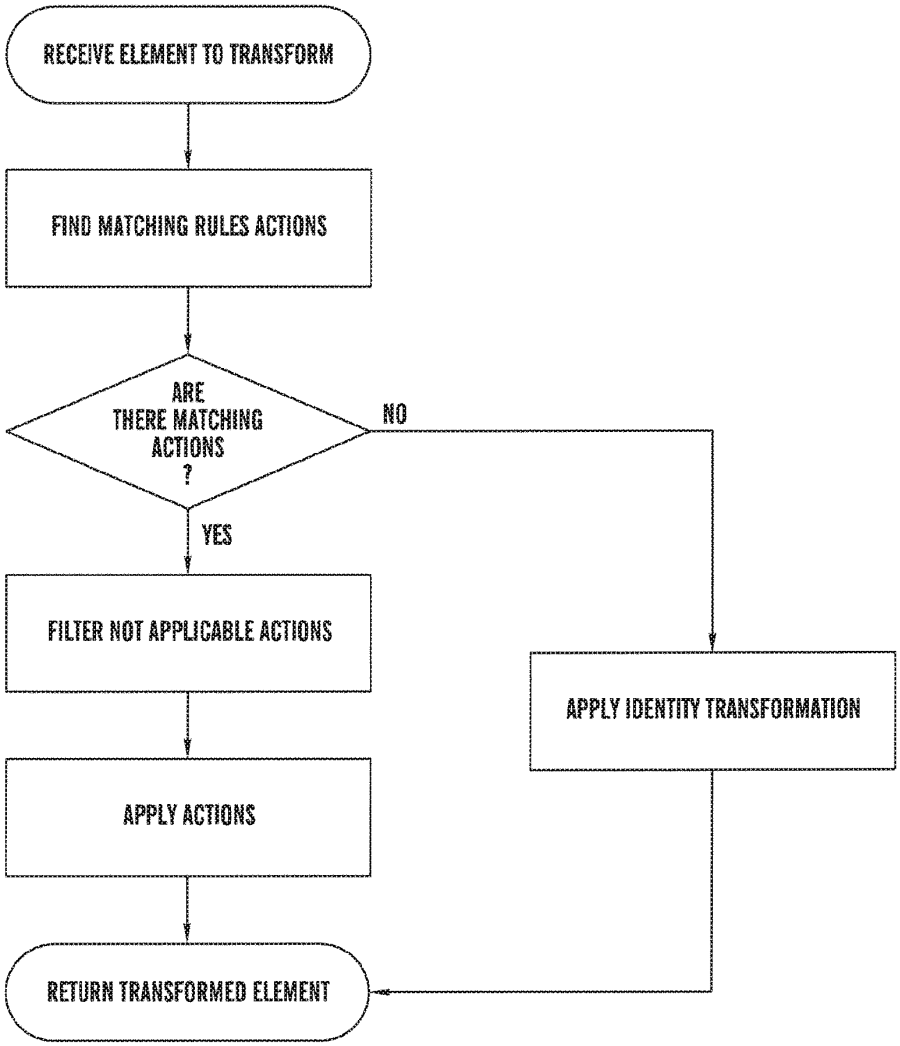

RECEIVE ELEMENT TO TRANSFORM
FIND MATCHING RULES ACTIONS
ARE THERE MATCHING ACTIONS ?
NO
YES
FILTER NOT APPLICABLE ACTIONS
APPLY IDENTITY TRANSFORMATION
APPLY ACTIONS
RETURN TRANSFORMED ELEMENT

FIG. 3

```
<rules>

    <rule for = "a">
        <match>@target='_blank'</match>
        <execute>
            <set-attribute name = "class">external</set-attribute>
            <remove-attribute name = "target"/>
        </execute>
    </rule>

    <rule for = "a">
        <match>img</match>
        <execute>
            <set-content>
                <xsl:value-of, select = "img/@alt"/>
            </set-content>
        </execute>
    </rule>

    <rule for ="a">
        <match>true()</match>
        <execute>
            <set-attribute name = "class">internal</set-attribute>
            <set-attribute name = "href" select = "myext:normalize-url(@href)"/>
        </execute>
    </rule>

</rules>
```

FIG. 4

```
<?xml version = "1.0" encoding = "UTF-8?>
<xsl:stylesheet xmln:xsl="http://ww.w3.org/1999/XSL/Transform" version="1.0"

    <xsl:template match="a">
        <xsl:copy>
        <xsl:attribute name="href"><xsl:value-of select-"myext:normalize-
        url(@href)"/></xsl:attribute>
        <xsl:choose>
            <xsl:when test = "@target='_blank">
                <xsl:attribute name="class">external</xsl:attribute>
            </xsl:when>
            <xsl:otherwise>
                <xsl:attribute name = "class">internal</xsl:attribute>
                <xsl:if test="@target">
                    <xsl:copy-of select="@target"/>
                </xsl:if>
            </xsl:otherwise>
        </xsl:choose>
        <xsl:choose>
            <xsl:when test = "img">
                <xsl:value-of select = "img/@alt"/>
            </xsl:when>
            <xsl:otherwise>
                <xsl:apply-templates/>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:copy>
</xsl:template>

    <xsl:template match-"*">
        <xsl:copy>
            <xsl:copy-of select="@*"/>
            <xsl:apply-templates/>
        </xsl:copy>
    </xsl:template>

</xsl:stylesheet>
```

FIG. 5

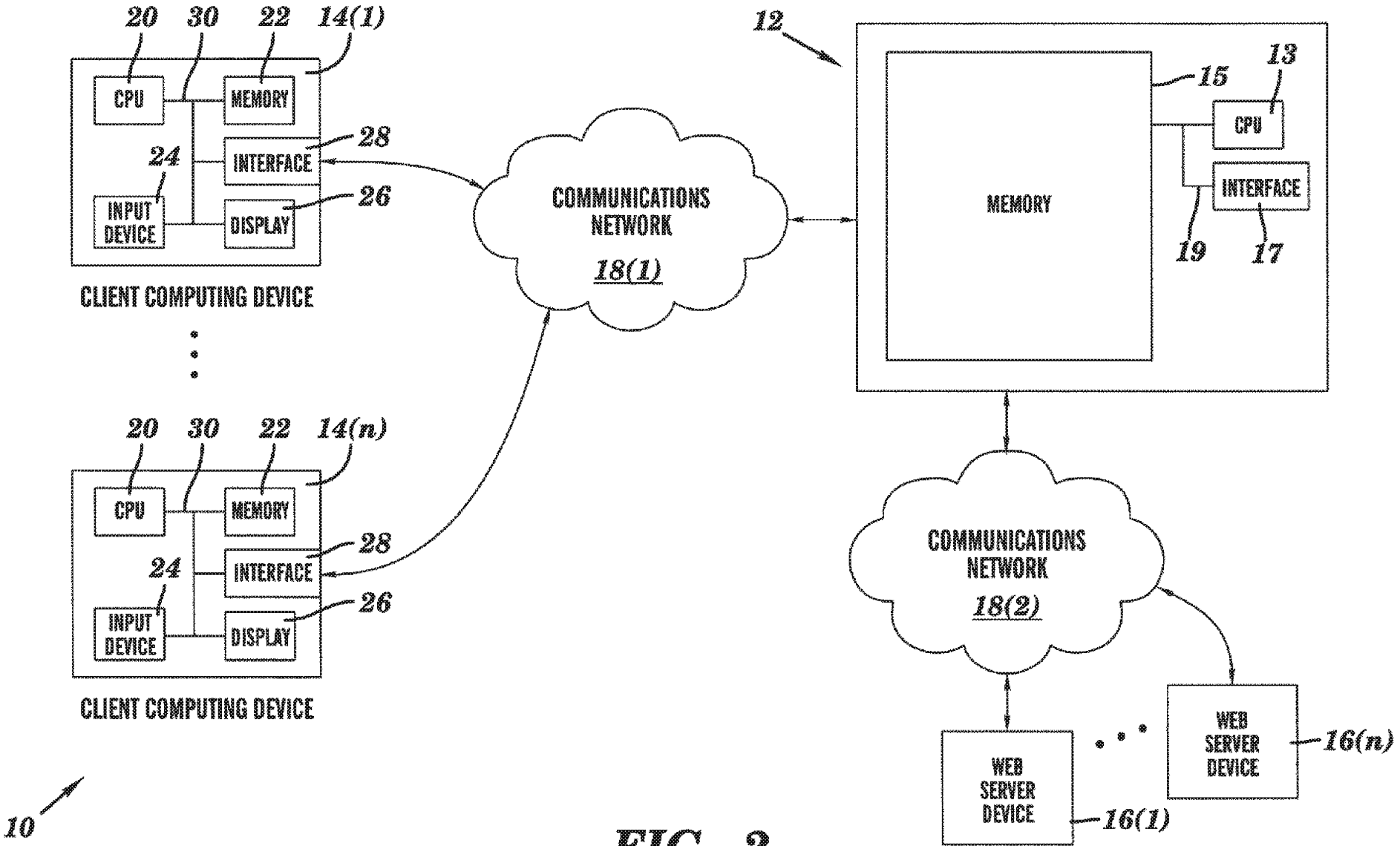


FIG. 2

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2811207 A1 2012/03/22

(21) 2 811 207

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(86) Date de dépôt PCT/PCT Filing Date: 2011/08/17
(87) Date publication PCT/PCT Publication Date: 2012/03/22
(85) Entrée phase nationale/National Entry: 2013/03/12
(86) N° demande PCT/PCT Application No.: US 2011/048060
(87) N° publication PCT/PCT Publication No.: 2012/036833
(30) Priorité/Priority: 2010/09/14 (US12/807,766)

(51) Cl.Int./Int.Cl. *G06F 17/21* (2006.01), *G06F 17/00* (2006.01)

(71) Demandeur/Applicant:
USABLENET, INC., US

(72) Inventeur/Inventor:
SCODA, ENRICO, IT

(74) Agent: GOWLING LAFLEUR HENDERSON LLP

(54) Titre : PROCEDES PERMETTANT D'ETENDRE UN SERVEUR DE TRANSFORMATION DE DOCUMENTS POUR LE TRAITEMENT DE PLUSIEURS DOCUMENTS PROVENANT DE PLUSIEURS SITES ET DISPOSITIFS ASSOCIES

(54) Title: METHODS FOR EXTENDING A DOCUMENT TRANSFORMATION SERVER TO PROCESS MULTIPLE DOCUMENTS FROM MULTIPLE SITES AND DEVICES THEREOF

(57) Abrégé/Abstract:
A method, computer readable medium and device that extends a document transformation server to process multiple documents from multiple websites includes obtaining with a document transformation server a document to process based on at least a URL of the document to process in an HTTP request URL. An XML source representing the document to process is generated with the document transformation server. One or more XML processors associated with at least a portion of the HTTP request URL from a plurality of stored associations are identified with the document transformation server. The XML source is transformed by the document transformation server with each of the identified one identified XML processors. The generated document is provided in an HTTP response to the HTTP request URL by the document transformation server.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

REVISED VERSION

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
22 March 2012 (22.03.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/036833 A9

(51) International Patent Classification:
G06F 17/21 (2006.01) *G06F 17/00* (2006.01)

(21) International Application Number:
PCT/US2011/048060

(22) International Filing Date:
17 August 2011 (17.08.2011)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/807,766 14 September 2010 (14.09.2010) US

(71) Applicant *(for all designated States except US)*: **US-ABLENET, INC.** [US/US]; 28 W. 23rd St., 6th Floor, New York, NY 10010 (US).

(72) Inventor: **SCODA, Enrico**; Via Cividina 416/3, I-33035 Martignacco (UD) (IT).

(74) Agents: **LEINBERG, Gunnar, G.** et al.; LeClairRyan, P.C., 70 Linden Oaks, Suite 210, Rochester, NY 14625 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

---- *with international search report (Art. 21(3))*

(88) Date of publication of the international search report:
9 August 2012

Date of publication of the revised international search report:
20 December 2012

(15) Information about Correction:
see Notice of 20 December 2012

(54) Title: METHODS FOR EXTENDING A DOCUMENT TRANSFORMATION SERVER TO PROCESS MULTIPLE DOCUMENTS FROM MULTIPLE SITES AND DEVICES THEREOF

(57) Abstract: A method, computer readable medium and device that extends a document transformation server to process multiple documents from multiple websites includes obtaining with a document transformation server a document to process based on at least a URL of the document to process in an HTTP request URL. An XML source representing the document to process is generated with the document transformation server. One or more XML processors associated with at least a portion of the HTTP request URL from a plurality of stored associations are identified with the document transformation server. The XML source is transformed by the document transformation server with each of the identified one identified XML processors. The generated document is provided in an HTTP response to the HTTP request URL by the document transformation server.

WO 2012/036833 A9

METHODS FOR EXTENDING A DOCUMENT TRANSFORMATION SERVER TO PROCESS MULTIPLE DOCUMENTS FROM MULTIPLE SITES AND DEVICES THEREOF

FIELD

[0001] This technology generally relates to methods and devices for transforming and rendering documents and, more particularly, to methods for extending a document transformation server to process multiple documents from multiple sites and devices thereof.

BACKGROUND

[0002] The introduction of eXtensible Markup Language (XML) and the EXtensibleStylesheet Language (XSL) specifications has provided an easy way to transform documents between various formats. This functionality has been included into Web development frameworks enabling them to automatically transform an XML file into a document with different format, such as HTML or XHTML, and integrate the original data with a graphic layout and user interface parts.

[0003] This transformation process is based on the following basic scheme. Each URL on a website is associated to a single XML file and to one or more XSL style sheets. The associated XML file and XSL style sheet(s) include the transformation rules used to customize the webpage into a desired format.

[0004] An example of a prior art method for transforming and rendering a single document is illustrated in FIG. 2. In step 100, a server receives an HTTP request. In step 102, the server extracts the parts from the PATH and QUERY fields in the HTTP request to identify and obtain the document to be processed.

[0005] In step 104, the server determines whether the extracted parts in the PATH and QUERY fields of the HTTP request match an XML source for the document to be processed. If in step 104 the server determines the extracted parts in the PATH and QUERY fields of the HTTP request do not have an association with an XML source, then the No branch is taken to step 106. In step 106, the

server generates an error page document. In step 108 the server returns the generated error page document in an HTTP response to the HTTP request.

[0006] If in step 104 the server determines the extracted parts in the PATH and QUERY fields of the HTTP request do have an association with an XML source for the document to be processed, then the Yes branch is taken to step 110. In step 110, the server obtains the document to be processed from the XML source and generates an XML document.

[0007] In step 112, the server determines whether the extracted parts in the PATH and QUERY fields of the HTTP request match one of one or more stored XML processors. If in step 112 the server determines the extracted parts in the PATH and QUERY fields of the HTTP request do not match one of one or more stored XML processors, then the No branch is taken to step 114. In step 114, the server converts the XML document into a desired format. Next, in step 108 the returns the converted XML document in an HTTP response to the HTTP request.

[0008] If in step 112 the server determines the extracted parts in the PATH and QUERY fields of the HTTP request do match one or more stored of the XML processors, then the Yes branch is taken to step 116. In step 116, the server runs the one or more matching XML processors on the previously generated XML document. In step 114, the server converts the processed XML document into a desired format. Next, in step 108 the server returns the converted XML document in an HTTP response to the HTTP request.

[0009] Referring to FIG. 3, an exemplary prior art fragment of a sitemap.xmap document which is a configuration file of Apache Cocoon is illustrated. This fragment has two sets of prior art instructions to process documents.

[00010] The first set of instructions executed starts with XML element <map:match pattern=""> and matches the document with an empty component in the PATH field that is the home page of a site (for example www.sample.com). The document is obtained by loading the XML file "welcome.xml" as the

document or other file to be processed, then applying the XSL transformation described in welcome.xsl using XHTML format to return it to the browser because of the <map:serialize> instruction.

[00011] The second set of instructions matches an entire set of web site pages: all pages whose PATH field starts with "static-site/" and ends with a name followed by ".xml" extension (for example, www.sample.com/static-site/news.xml). The first instruction loads the corresponding documents or other files from the xdocs directory (for example xdocs/news.xml). The second instruction applies an XSL transformation using one or more identified matching XML processors that transforms the original xml document into an HTML document adding web site user interface (web site navigation links, site logo etc). The last instruction returns the document to the browser in HTML format.

[00012] Accordingly, as illustrated and described above, the prior art provides methods for transforming and rendering documents. Unfortunately, these transformation rules currently can not be utilized in an effective and efficient manner by a document transformation server in a single application instance to transform multiple documents from multiple websites.

SUMMARY

[00013] A method for extending a document transformation server to process multiple documents from multiple websites includes obtaining with a document transformation server a document to process based on at least a URL of the document to process in an HTTP request URL. An XML source representing the document to process is generated with the document transformation server. One or more XML processors associated with at least a portion of the HTTP request URL from a plurality of stored associations are identified with the document transformation server. The XML source is transformed by the document transformation server with each of the identified one identified XML processors. The generated document is provided in an HTTP response to the HTTP request URL by the document transformation server.

[00014] A non-transitory computer readable medium having stored thereon instructions for extending a document transformation server to process multiple documents from multiple websites comprising machine executable code which when executed by at least one processor, causes the processor to perform steps including obtaining a document to process based on at least a URL of the document to process in an HTTP request URL. An XML source representing the obtained document to process is generated and one or more XML processors that are associated with at least a portion of the HTTP request URL from a plurality of stored associations are identified. The XML source is transformed with each of the identified one identified XML processors and the generated document is provided in an HTTP response to the HTTP request URL.

[00015] A document transformation apparatus includes one or more processors and a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory obtaining a document to process based on at least a URL of the document to process in an HTTP request URL. An XML source representing the obtained document to process is generated and one or more XML processors that are associated with at least a portion of the HTTP request URL from a plurality of stored associations are identified. The XML source is transformed with each of the identified one identified XML processors and the generated document is provided in an HTTP response to the HTTP request URL.

[00016] This technology provides a number of advantages including providing a method, computer readable medium and an apparatus that extends a document transformation server to process multiple documents from multiple sites. With this technology, an exemplary document transformation server can efficiently manage the transformation and rendering of documents using a single web application instance.

BRIEF DESCRIPTION OF THE DRAWINGS

[00017] FIG. 1 is a block diagram of an exemplary environment with an exemplary document transformation server configured to process multiple documents from multiple sites;

[00018] FIG. 2 is a flow chart of a prior art method for transforming and rendering a document;

[00019] FIG. 3 is a prior art fragment of a configuration file;

[00020] FIG. 4 is an exemplary flow chart of a method for extending a document transformation server to process multiple documents from multiple sites;

[00021] FIG. 5 is an exemplary flow chart of a method for obtaining a mapping file; and

[00022] FIG. 6 is an exemplary implementation of a mapping file.

DETAILED DESCRIPTION

[00023] An exemplary environment 10 with a document transformation server 12 configured to process multiple documents from multiple websites is illustrated in FIG. 1. The exemplary environment 10 includes the document transformation server or apparatus 12, client devices 14(1)-14(n), web server devices 16(1)-16(n), and communication networks 18(1)-18(2), although other numbers and types of systems, devices, and/or elements in other configurations and environments with other communication network topologies can be used. This technology provides a number of advantages including providing a method, computer readable medium and an apparatus that extends a document transformation server to process multiple documents from multiple sites.

[00024] Referring more specifically to FIG. 1, the document transformation server 12 includes a central processing unit (CPU) or processor 13, a memory 15, and an interface system 17 which are coupled together by a bus 19

or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. Additionally, other types and numbers of proxy servers or other computing devices could be configured to execute the exemplary methods illustrated and described herein. The processor 13 in the document transformation server 12 executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[00025] The memory 15 in the document transformation server 12 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor 13, can be used for the memory 15 in the document transformation server 12.

[00026] The interface system 17 in the document transformation server 12 is used to operatively couple and communicate between the document transformation server 12 and the client devices 14(1)-14(n) and the web server devices 16(1)-16(n) via the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used. By way of example only, the communication networks 18(1) and 18(2) can use TCP/IP over Ethernet and industry-standard protocols, including HTTP, HTTPS, WAP, and SOAP, although other types and numbers of communication networks, such as a direct connection, a local area network, a wide area network, modems and phone lines, e-mail, and wireless and hardwire communication technology, each having their own communications protocols, can be used.

[00027] Each of the client devices 14(1)-14(n) enables a user to request, get and interact with documents and other files from one or more web sites hosted by the web server devices 16(1)-16(n) through the document transformation server 12 via one or more communication networks, although one or more of the client devices 14(1)-14(n) could access content and utilize other types and numbers of applications from other sources and could provide a wide variety of other functions for the user. Although multiple client devices 14(1)-14(n) are shown, other numbers and types of user computing systems could be used.

[00028] Each of client devices 14(1)-14(n) in this example is a computing device that includes a central processing unit (CPU) or processor 20, a memory 22, user input device 24, a display 26, and an interface system 28, and which are coupled together by a bus 30 or other link, although one or more of client devices 14(1)-14(n) can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 20 in each of client devices 14(1)-14(n) executes a program of stored instructions for one or more aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions.

[00029] The memory 22 in each of the client devices 14(1)-14(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor 20 can be used for the memory 22 in each of the client devices 14(1)-14(n.

[00030] The user input device 24 in each of the client devices 14(1)-14(n) is used to input selections and other data, although the user input device could provide other functions and interact with other elements. The user input device

can include keypads, touch screens, and/or vocal input processing systems although other types and numbers of user input devices can be used.

[00031] The display 26 in each of the client devices 14(1)-14(n) is used to show data and information to the user, such as a website page by way of example only. The display in each of the client devices 14(1)-14(n) is a computer screen display, although other types and numbers of displays could be used depending on the particular type of client device.

[00032] The interface system 28 in each of the client devices 14(1)-14(n) is used to operatively couple and communicate between the client devices 14(1)-14(n) and the document transformation server 12 and web server devices 16(1)-16(n) over the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00033] The web server devices 16(1)-16(n) provide one or more pages from one or more web sites for use by one or more of the client devices 14(1)-14(n) via the document transformation server 12, although the web server devices 16(1)-16(n) can provide other numbers and types of applications and/or content and can have provide other numbers and types of functions. Although web server devices 16(1)-16(n) are shown for ease of illustration and discussion, other numbers and types of web server systems and devices can be used.

[00034] Each of the web server devices 16(1)-16(n) include a central processing unit (CPU) or processor, a memory, and an interface system which are coupled together by a bus or other link, although each of the web server devices 16(1)-16(n) could have other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor in each of the web server devices 16(1)-16(n) executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[00035] The memory in each of the web server devices 16(1)-16(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated by way of the embodiments, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor, can be used for the memory in each of the web server devices 16(1)-16(n).

[00036] The interface system in each of the web server devices 16(1)-16(n) is used to operatively couple and communicate between the web server devices 16(1)-16(n) and the document transformation server 12 and the client devices 14(1)-14(n) via communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00037] Although embodiments of the document transformation server 12, the client devices 14(1)-14(n), and the web server devices 16(1)-16(n), are described and illustrated herein, each of the client devices 14(1)-14(n), the document transformation server 12, and the web server devices 16(1)-16(n), can be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s).

[00038] Furthermore, each of the systems of the embodiments may be conveniently implemented using one or more general purpose computer systems, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[00039] In addition, two or more computing systems or devices can be substituted for any one of the systems in any embodiment of the embodiments. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the embodiments. The embodiments may also be implemented on computer system or systems that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[00040] The embodiments may also be embodied as non-transitory computer readable medium having instructions stored thereon for one or more aspects of the present invention as described and illustrated by way of the embodiments herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the embodiments, as described and illustrated herein.

[00041] An exemplary method for extending a document transformation server 12 to process multiple documents from multiple sites, such as web server devices 16(1)-16(n), in an exemplary environment 10 will now be described with reference to FIGS. 1 and 4-6. Referring more specifically to FIG. 4, in step 200 the document transformation server 12 receives an HTTP request whose url contains the URI of the document to process, from one of the client computing devices 14(1)-14(n), although other types of requests could be received and from other types of devices. Optionally, if the URI of the document to process is encoded, the document transformation server 12 will decode the URI of the document to process, although other manners for processing an encoded URI or other encoded portion of a URI HTTP request could be used. Since manners for encoding and decoding are well known to those of ordinary skill in the art, they will not be described in detail here. By way of example only, consider an HTTP

request from one of the client computing devices 14(1)-14(n) where the URL to be processed is embedded in the PATH field of the HTTP request url: http://processor.com/app/www.acme.com/anyproduct.html. The transformer server applications is identified by "processor.com/app"; the address www.acme.com/anyproduct.html is the resource to be processed.

[00042] In step 202, the document transformation server 12 extracts the domain in the URI of the document to process, although other fields of the URI could be extracted and used.

[00043] In step 204, the document transformation server 12 determines whether the transformation of the website in the URI of the document to process is allowed. If in step 204, the document transformation server 12 determines the transformation of the website in the URI of the document to process is not allowed, then the No branch is taken to step 206. In step 206, the document transformation server 12 generates an error page document. In step 208 the document transformation server 12 returns the generated error page document in an HTTP response to the one of the client computing devices 14(1)-14(n) that provided the HTTP request in this example.

[00044] If in step 204, the document transformation server 12 determines the transformation of the website in the URI of the document to process is allowed, then the Yes branch is taken to step 210. In step 210, the document transformation server 12 downloads the original document or other resource linked by the given URL and transforms the downloaded document into an XML source, although other manners for processing the obtained document or documents can be used. In step 212, document transformation server 12 looks for a mapping or configuration file for the directory that matches the extracted domain. An exemplary method for locating a mapping file from manifest.xml is described herein with reference to FIG. 5.

[00045] In step 214, the document transformation server 12 determines whether the mapping file for the extracted domain exists. If in step 214 the document transformation server 12 determines the mapping file does not exist,

then the No branch is taken to step 216. In step 216, the document transformation server 12 converts the document into the desired format. Next, in step 208 the document transformation server 12 returns the converted document in an HTTP response to the one of the client computing devices 14(1)-14(n) that provided the HTTP request.

[00046] If in step 214 the document transformation server 12 determines the mapping file for the extracted domain does exist, then the Yes branch is taken to step 218. In step 218, the document transformation server 12 use the mapping file to identify all of the XSL processors linked to the corresponding URL and applies the identified XSL processors to the document to be processed. Optionally the document transformation server 12 may apply one or more best matching rules to filter the identified one or more XML processors in the matching directory. By way of example only, one best matching rule is to take the XML processor with the longest matching pattern string, although other types and numbers of best matching rules can be used.

[00047] In step 216, the document transformation server 12 converts the processed document into the desired format. Next, in step 208 the document transformation server returns the converted document in an HTTP response to the one of the client computing devices 14(1)-14(n) that provided the HTTP request.

[00048] Referring to FIG. 5, an exemplary method for obtaining a mapping file with the document transformation server 12 is illustrated. In step 300, the document transformation server 12 receives a URI to match from one of the client computing devices 14(1)-14(n), although other types of requests could be received and from other types of devices. In step 302, the document transformation server 12 extracts from the URI the AUTHORITY portion corresponding to the associated domain and port, for example "ssl.example.com:8443", although other types and numbers of fields could be extracted.

[00049] In this example, if the port is either 80 or 443 (default values for HTTP and HTTPS respectively) the string representing the authority corresponds to the domain portion of the URL. However, if the port does not correspond to a

standard value, the ':' character is substituted with '_' in order to avoid potential problems in the file system (i.e. ssl.example.com_8443). The string representing the authority is further stripped by the document transformation server 12 removing the 'www.' from the extracted string.

[00050] In step 304, the document transformation server 12 determines whether there is a directory with the same name as the extracted domain and containing the mapping file, although other manners for determining matches with other portions of the domain can be used. By way of example only, consider the following directories: (1) "www.acme.com"; (2) "_.example.com"; and (3) "ssl.example.com". The www.acme.com directory will match all the requests for www.acme.com and acme.com domains. With this technology, the "_.example.com" directory will match all the requests for domains ending with ".example.com" that do not have more specific matching directories. Accordingly, "_.example.com" will match "products.example.com", but it will not match "ssl.example.com" because an "ssl.example.com" directory exists. The "_" character in front of ".example.com" is used to make the directory visible when using file systems that use "." characters in front of files to hide them to users.

[00051] If in step 304 the document transformation server 12 determines there is no directory with the same name as the extracted domain, then the No branch is taken to step 306. In step 306, the document transformation server 12 determines whether the domain contains a valid subdomain. If in step 306 the document transformation server 12 determines the domain does contain a valid subdomain, then the Yes branch is taken to step 308. In step 308, the document transformation server 12 extracts the next level subdomain name and returns to step 304 as described earlier.

[00052] If in step 304 the document transformation server 12 determines there is a directory with the same name as the extracted domain and containing the mapping file, then the Yes branch is taken to step 310, although other manners for determining matches with other portions of the domain can be used. In another example, the document transformation server 12 may also use another optional mapping file that stores known aliases of domains. When a known alias of a

domain is identified using this optional mapping file, the document transformation server 12 can use the same one or more stored directories for the known and identified aliases. As a result, this reduces the number of stored directories when handling domains with multiple aliases (i.e. defined with DNS records of type CNAME).

[00053] In step 310, the document transformation server 12 extracts the parts from the PATH and QUERY fields in the URI, although the parts can be extracted from other types and numbers of fields in the URI.

[00054] In step 312, the document transformation server 12 determines whether the extracted parts in the PATH and QUERY fields of the URI match or otherwise have an association with one of one or more XML processors in the identified mapping file, although other types of associations between other types and numbers of parts can be used. If in step 312 the document transformation server 12 determines the extracted parts in the PATH and QUERY fields of the URI match or otherwise have an association with one of one or more XML processors, then the Yes branch is taken to step 314 where the matching XML processors are collected. Next, in step 316, the document transformation server 12 returns the collected XML processors to the requesting one of the client computing devices 14(1)-14(n).

[00055] If in step 312 the document transformation server 12 determines the extracted parts in the PATH and QUERY fields of the URI do not match one of one or more XML processors, then the No branch is taken to step 316 where no matching XML processors are returned to the requesting one of the client computing devices 14(1)-14(n).

[00056] Referring to FIG. 6, an exemplary implementation of the mapping file "manifest.xml" is illustrated. Each map element identifies an XSL file (inside file element) and a substring to match a URL PATH component (inside the PATH element). The map element with the longest string matching the PATH component will be used to setup the XSL transformer identified by the level attribute.

[00057] In these examples, two XSL transformers are defined: level 1 transformer; and level 2 transformer. If the request URL has the PATH field containing "/content/news/july.html", then the original site web page is downloaded (for example, www.sample.com/content/news/july.html). Next, the level 1 transformer uses the news.xsl stylesheet document to transform the downloaded page. The level 2 transformer uses content2.xsl stylesheet document to transform the document obtained from the level 1 transformer.

[00058] If the request URL has the PATH field containing "/content/privacy.html", then the original site web page is downloaded (for example www.sample.com/content/privacy.html). Next, the level 1 transformer uses the content.xsl stylesheet document to transform the downloaded page. Next, the level 2 transformer uses the content2.xsl stylesheet document to transform the document obtained from the level 1 transformer.

[00059] Accordingly, as illustrated and described herein this technology provides a number of advantages including providing a method, computer readable medium and an apparatus that extends a document transformation server to process multiple documents from multiple sites. One of the advantages of this technology is that a single association or match between a name of a directory and at least a portion of a domain name or other part of a URI can be used for hundreds or thousands of URIs. As a result, only a few associations with directories or other stored tables can be enough for the document transformation server to transform the content of an entire website.

[00060] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order

except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for extending a document transformation server to process multiple documents from multiple websites, the method comprising:

obtaining with a document transformation server a document to process based on at least a URL of the document to process in an HTTP request URL;

generating with the document transformation server an XML source representing the document to process;

identifying with the document transformation server one or more XML processors associated with at least a portion of the HTTP request URL from a plurality of stored associations;

transforming with the document transformation server the XML source with each of the identified one identified XML processors; and

providing with the document transformation server the generated document in an HTTP response to the HTTP request URL.

2. The method as set forth in claim 1 wherein the obtaining the document to process is further based on one or more additional identifiers in the HTTP request URL.

3. The method as set forth in claim 2 wherein the one or more additional identifiers comprise at least one of an HTTP header and an HTTP POST parameter.

4. The method as set forth in claim 1 wherein the obtaining the document to process further comprises decoding with the document transformation server the obtained URL of the document to process prior to obtaining the document to process.

5. The method as set forth in claim 1 further comprising:

storing with the document transformation server the one or

more XML processors associated to at least a portion of a particular domain in a directory having a directory name equal to the at least a portion of the particular domain; and

storing with the document transformation server a mapping file between the one or more XML processor and one or more parts of one or more fields in an HTTP request URL with the stored directory.

6. The method as set forth in claim 5 wherein the one or more fields of the HTTP requests URL comprise one or more of a URL path, one or more query parameters, one or more post parameters, and one or more HTTP headers.

7. The method as set forth in claim 6 wherein the identifying with the document transformation server the one or more XML processors further comprises:

locating with the document transformation server the directory comprising one or more XML processors which has a directory name equal to at least a portion of the particular domain in the HTTP request URL;

opening with the document transformation server the mapping file with the located directory comprising the mapping file between the one or more XML processor and one or more parts of one or more fields in an HTTP request URL; and

identifying with the document transformation server the one or more XML processors in the located directory that match the one or more parts of the one or more fields in an HTTP request URL and filtering out any non-matching XML processors.

8. The method as set forth in claim 7 further comprising applying with the document transformation server one or more best match rules to the identified one or more XML processors, wherein the transforming with the document transformation server the XML source is based on the identified best matches of the identified one or more XML processors.

9. The method as set forth in claim 7 wherein the locating with the document transformation server the directory further comprises:

determining with the document transformation server when more than one directory has a directory name equal to at least a portion of the particular domain in the HTTP request URL; and

selecting with the document transformation server the directory which has a name which is the longest among the located directories as the located directory.

10. The method as set forth in claim 1 wherein each of the one or more XML processors comprises a set of one or more XSL transformers with transformation rules in one or more XSL files.

11. A non-transitory computer readable medium having stored thereon instructions for extending a document transformation server to process multiple documents from multiple websites comprising machine executable code which when executed by at least one processor, causes the processor to perform steps comprising:

obtaining a document to process based on at least a URL of the document to process in an HTTP request URL;

generating an XML source representing the document to process;

identifying one or more XML processors associated with at least a portion of the HTTP request URL from a plurality of stored associations;

transforming the XML source with each of the identified one identified XML processors; and

providing the generated document in an HTTP response to the HTTP request URL.

12. The medium as set forth in claim 11 wherein the obtaining the document to process is further based on one or more additional identifiers in the HTTP request URL.

13. The medium as set forth in claim 12 wherein the one or more additional identifiers comprise at least one of an HTTP header and an HTTP POST parameter.

14. The medium as set forth in claim 11 wherein the obtaining the document to process further comprises decoding the obtained URL of the document to process prior to obtaining the document to process.

15. The medium as set forth in claim 11 further comprising:

storing the one or more XML processors associated to at least a portion of a particular domain in a directory having a directory name equal to the at least a portion of the particular domain; and

storing a mapping file between the one or more XML processor and one or more parts of one or more fields in an HTTP request URL with the stored directory.

16. The medium as set forth in claim 15 wherein the one or more fields of the HTTP requests URL comprise one or more of a URL path, one or more query parameters, one or more post parameters, and one or more HTTP headers.

17. The medium as set forth in claim 16 wherein the identifying the one or more XML processors further comprises:

locating the directory comprising one or more XML processors which has a directory name equal to at least a portion of the particular domain in the HTTP request URL;

opening the mapping file with the located directory comprising the mapping file between the one or more XML processor and one or more parts of one or more fields in an HTTP request URL; and

identifying the one or more XML processors in the located directory that match the one or more parts of the one or more fields in an HTTP request URL and filtering out any non-matching XML processors.

18. The medium as set forth in claim 17 further comprising applying one or more best match rules to the identified one or more XML processors, wherein the transforming the XML source is based on the identified best matches of the identified one or more XML processors.

19. The medium as set forth in claim 17 wherein the locating the directory further comprises:

determining when more than one directory has a directory name equal to at least a portion of the particular domain in the HTTP request URL; and

selecting the directory which a name which is the longest among the located directories as the located directory.

20. The medium as set forth in claim 11 wherein each of the one or more XML processors comprises a set of one or more XSL transformers with transformation rules in one or more XSL files.

21. A document transformation apparatus comprising:

one or more processors;

a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory comprising:

obtaining a document to process based on at least a URL of the document to process in an HTTP request URL;

generating an XML source representing the document to process;

identifying one or more XML processors associated with at least a portion of the HTTP request URL from a plurality of stored associations;

transforming the XML source with each of the identified one identified XML processors; and

providing the generated document in an HTTP response to the HTTP request URL.

22. The apparatus as set forth in claim 21 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for the obtaining the document to process further comprising obtaining the document to process further based on one or more additional identifiers in the HTTP request URL.

23. The apparatus as set forth in claim 22 wherein the one or more additional identifiers comprise at least one of an HTTP header and an HTTP POST parameter.

24. The apparatus as set forth in claim 21 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising decoding the obtained URL of the document to process prior to obtaining the document to process.

25. The apparatus as set forth in claim 21 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising:

storing the one or more XML processors associated to at least a portion of a particular domain in a directory having a directory name equal to the at least a portion of the particular domain; and

storing a mapping file between the one or more XML processor and one or more parts of one or more fields in an HTTP request URL with the stored directory.

26. The apparatus as set forth in claim 25 wherein the one or more fields of the HTTP requests URL comprise one or more of a URL path, one or more query parameters, one or more post parameters, and one or more HTTP headers.

27. The apparatus as set forth in claim 26 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for the identifying the one or more XML processors further

comprising:

locating the directory comprising one or more XML processors which has a directory name equal to at least a portion of the particular domain in the HTTP request URL;

opening the mapping file with the located directory comprising the mapping file between the one or more XML processor and one or more parts of one or more fields in an HTTP request URL; and

identifying the one or more XML processors in the located directory that match the one or more parts of the one or more fields in an HTTP request URL and filtering out any non-matching XML processors.

28. The apparatus as set forth in claim 27 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising applying one or more best match rules to the identified one or more XML processors, wherein the transforming the XML source is based on the identified best matches of the identified one or more XML processors.

29. The apparatus as set forth in claim 27 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for the locating the directory further comprising:

determining when more than one directory has a directory name equal to at least a portion of the particular domain in the HTTP request URL; and

selecting the directory which has a name which is the longest among the located directories as the located directory.

30. The apparatus as set forth in claim 21 wherein each of the one or more XML processors comprises a set of one or more XSL transformers with transformation rules in one or more XSL files.

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2816336 A1 2012/05/18

(21) 2 816 336

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(86) Date de dépôt PCT/PCT Filing Date: 2011/11/09
(87) Date publication PCT/PCT Publication Date: 2012/05/18
(85) Entrée phase nationale/National Entry: 2013/04/26
(86) N° demande PCT/PCT Application No.: US 2011/059992
(87) N° publication PCT/PCT Publication No.: 2012/064856
(30) Priorité/Priority: 2010/11/09 (US12/927,169)

(51) Cl.Int./Int.Cl. *H04L 29/06* (2006.01),
G06F 15/16 (2006.01)

(71) Demandeur/Applicant:
USABLENET INC., US

(72) Inventeur/Inventor:
SCODA, ENRICO, IT

(74) Agent: GOWLING LAFLEUR HENDERSON LLP

(54) Titre : PROCEDES DE REDUCTION DE LATENCE DANS DES CONNEXIONS DE RESEAU ET SYSTEMES CORRESPONDANTS

(54) Title: METHODS FOR REDUCING LATENCY IN NETWORK CONNECTIONS AND SYSTEMS THEREOF

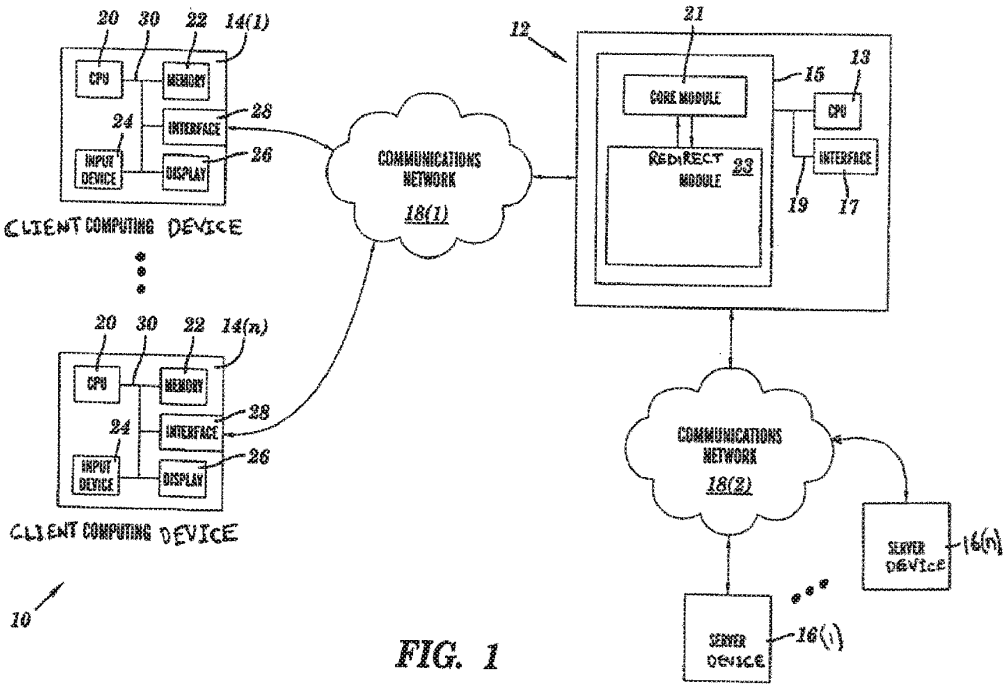


FIG. 1

(57) Abrégé/Abstract:
A method, computer readable medium, and apparatus for reducing latency in network connections include receiving at a proxy server device a first response from one or more server devices in response to a first request from a client device for a network resource stored on the one or more server devices. The proxy server device determines whether or not the first response is a redirect message, and sends a second request to the one or more server devices when the first response includes the first redirect message, such that the sending is performed at least partially based upon the received redirect message.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
18 May 2012 (18.05.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/064856 A3

(51) International Patent Classification:
H04L 29/06 (2006.01) *G06F 15/16* (2006.01)

(21) International Application Number:
PCT/US2011/059992

(22) International Filing Date:
9 November 2011 (09.11.2011)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/927,169 9 November 2010 (09.11.2010) US

(71) Applicant *(for all designated States except US)*: **US-ABLENET INC.** [US/US]; 28 W. 23rd Street, 6th Floor, New York, NY 10010 (US).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: **SCODA, Enrico** [IT/IT]; Via Cividina 416/3, Martignacco (ud) (IT).

(74) Agents: **LEINBERG, Gunnar, G.** et al.; Leclairryan, A professional Corporation, 70 Linden Oaks, Suite 210, Rochester, NY 14625 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(88) Date of publication of the international search report:
4 October 2012

(54) Title: METHODS FOR REDUCING LATENCY IN NETWORK CONNECTIONS AND SYSTEMS THEREOF

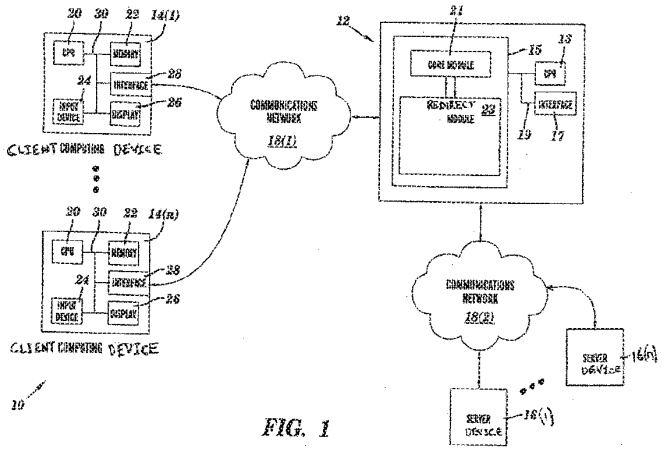


FIG. 1

(57) Abstract: A method, computer readable medium, and apparatus for reducing latency in network connections include receiving at a proxy server device a first response from one or more server devices in response to a first request from a client device for a network resource stored on the one or more server devices. The proxy server device determines whether or not the first response is a redirect message, and sends a second request to the one or more server devices when the first response includes the first redirect message, such that the sending is performed at least partially based upon the received redirect message.

WO 2012/064856 A3

METHODS FOR REDUCING LATENCY IN NETWORK CONNECTIONS AND SYSTEMS THEREOF

FIELD

[0001] The embodiments of the present invention generally relate to proxy server devices and, more particularly, to methods for reducing latency in network connections utilizing proxy server devices, and systems thereof.

BACKGROUND

[0002] When a client device connects to a server (*e.g.*, a web server, or a content server) to get a network resource using a network protocol, *e.g.*, the Hyper-text Transfer protocol (HTTP), the server responds by sending the network resource or by sending a redirect message back to the client device over a communication channel. If the client device receives a redirect message, it will need to send a new request to the server based upon the redirect message, and the server will again respond with a redirect or a real resource. This communication process between the client device and the server repeats until the client device is able to get the resource, if available.

[0003] However, when the client device, *e.g.*, a cell phone using a radio network, or a computer having a slow Internet connection, requests a network resource and has to perform more than one redirects to obtain the network resource, the client device will experience substantial delay and will spend a considerable amount of time to execute the whole process before finally being provided with the network resource. The delay can occur, for example, because of a large time to establish a connection and send the HTTP request, also referred to as latency time of radio networks or other slow network connections (*e.g.*, Internet via a dial-up connection). Unfortunately, this delay can often lead to the client device not being able to obtain the network resource at all, or the client device giving up or relinquishing attempts to obtain the network resource under time constraints.

[0004] One conventional solution built to obtain faster HTTP responses for slow connection networks utilizes one or more proxy server devices (*e.g.*, web proxy servers). Another conventional solution utilizes telephone carrier data centers that handle the network traffic for each client device by handling one or more requests when the client device is a mobile telephone, or a mobile personal digital assistant (PDA) device, for example. Unfortunately, the above-noted conventional solutions do not resolve the redirection problem associated with network resources stored on the servers since the redirect messages are forwarded on to the client devices for handling resulting in multiple back and forth communication between the client devices and the servers.

SUMMARY

[0005] A method for reducing latency in network connections includes receiving at a proxy server device a first response from one or more server devices in response to a first request from a client device for a network resource stored on the one or more server devices. The proxy server device determines whether or not the first response is a redirect message, and sends a second request to the one or more server devices when the first response includes the first redirect message, such that the sending is performed at least partially based upon the received redirect message.

[0006] A computer readable medium having stored thereon instructions for reducing latency in network connections comprising machine executable code which when executed by at least one processor, cause the processor to perform steps including receiving at a proxy server device a first response from one or more server devices in response to a first request from a client device for a network resource stored on the one or more server devices. The steps include determining at the proxy server device whether or not the first response is a redirect message, and sending a second request from the proxy server device to the one or more server devices when the first response includes the first redirect message, such that the sending is performed at least partially based upon the received redirect message.

[0007] An apparatus configured to reduce latency includes one or more processors and a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory including instructions to implement receiving at a proxy server device a first response from one or more server devices in response to a first request from a client device for a network resource stored on the one or more server devices. The instructions include instructions for determining at the proxy server device whether or not the first response is a redirect message, and sending a second request from the proxy server device to the one or more server devices when the first response includes the first redirect message, such that the sending is performed at least partially based upon the received redirect message.

[0008] This technology provides a number of advantages including providing a method, computer readable medium and an apparatus that adds an exemplary redirect module to the proxy server devices to efficiently manage the whole redirect chain returning only the last redirect message to the client device, which client device can then obtain the requested network resource from the server on which the network resource is stored, without exchanging multiple intermediate redirect messages back and forth with the server. In one exemplary scenario, cookies received by the proxy server device from the content servers are collected and forwarded to the client device. Accordingly, this technology provides substantial reduction in latency of network connections because the number of redirect messages between the client devices and the content/resource servers is reduced, and takes advantage of the high speed and high bandwidth communication infrastructure between the proxy server device(s) and the server devices on which content is stored.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] FIG. 1 is a block diagram of an exemplary network environment with a proxy server device interposed between client devices and server devices;

[00010] FIG. 2 is an example of a redirect response message chain exchanged between the client devices and the server devices of FIG. 1, as handled by a conventional proxy server device;

[00011] FIG. 3 is an example of an optimized redirect response message chain handled by the proxy server device of FIG. 1 on behalf of client devices of FIG. 1 for obtaining a network resource; and

[00012] FIG. 4 is an exemplary flowchart for reducing latency in network connections by optimizing redirect response message chains at the proxy server device.

DETAILED DESCRIPTION

[00013] An exemplary environment 10 in which a proxy server device 12 is optimized for reducing latency in network connections is illustrated in FIG. 1. By way of example only, proxy server device 12 can be a web content proxy server, or other types of proxy servers well known to those of ordinary skill in the art. The exemplary environment 10 includes the proxy server device or apparatus 12, client devices 14(1)-14(n), server devices 16(1)-16(n), and communication networks 18(1)-18(2), although other numbers and types of systems, devices, and/or elements in other configurations and environments with other communication network topologies can be used. This technology provides a number of advantages including providing a method, computer readable medium, and an apparatus that reduces latency in network connections, for example, HTTP connections.

[00014] Referring more specifically to FIG. 1, the proxy server device 12 manages handling of redirect messages or redirection responses from the server devices 16(1)-16(n) for and/or on behalf of requesting client devices 14(1)-14(n) and provides updated cookie information to the client devices 14(1)-14(n) for future requests for network resources from the client devices 14(1)-14(n), although the proxy server device 12 can provide other numbers and types of functions. Although one proxy server device 12 is shown, other numbers and types of web content proxy devices and systems can be used. By way of example

only, according to one embodiment of the present invention, proxy server device 12 can be a carrier gateway device communicating with one or more content servers, *e.g.*, server devices 16(1)-16(n), for managing network resource requests from client devices 14(1)-14(n). Further by way of example only, the proxy server device 12 can be an HTTP proxy server or a carrier gateway configured to reduce latency in network connections by receiving a first response from one or more server devices 16(1)-16(n) in response to a first request from a client device for a network resource stored on the one or more server devices 16(1)-16(n). Proxy server device 12 determines whether or not the first response is a redirect message, and sends a second request to the one or more server devices 16(1)-16(n) when the first response includes the first redirect message, such that the sending is performed at least partially based upon the received redirect message.

[00015] The proxy server device 12 includes a central processing unit (CPU) or processor 13, a memory 15, and an interface system 17 which are coupled together by a bus 19 or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor 13 in the proxy server device 12 executes a program of stored instructions to carry out or perform one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[00016] The memory 15 in the proxy server device 12 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor 13, can be used for the memory 15 in the proxy server device 12. In these embodiments, the memory 15 includes a core module 21 and a redirect module 23 which store programmed instructions for one or more aspects

of the present invention as described and illustrated herein, although the memory can comprise other types and numbers of systems, devices, and elements in other configurations which store other data. As discussed in more detail below in FIG. 3, only the last network location (*e.g.*, a Uniform Resource Identifier or URI) of a redirect chain 300 is forwarded by redirect module 23 to the requesting client device among client devices 14(1)-14(n), although the redirect module 23 can have other types and numbers of functions as described and illustrated herein.

[00017] The interface system 17 in the proxy server device 12 is used to operatively couple and communicate between the proxy server device 12 and the client devices 14(1)-14(n) and the server devices 16(1)-16(n) via the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used. By way of example only, the communication networks 18(1) and 18(2) can use TCP/IP over Ethernet and industry-standard protocols, including HTTP, HTTPS, WAP, and SOAP, although other types and numbers of communication networks, such as a direct connection, a local area network, a wide area network, modems and phone lines, e-mail, and wireless and hardwire communication technology, each having their own communications protocols, can be used. In one exemplary embodiment, one of communication networks 18(1) and 18(2) can be operating over one or more low-speed connections (*e.g.*, a dial-up connection) while the other one of the communication networks 18(1) and 18(2) can be operating over a high speed, high bandwidth connection (*e.g.*, optical fiber based communication network). In yet another exemplary embodiment, one or more of communication networks 18(1) and 18(2) can be a radio network, a satellite network, an Internet connection, a wired cable network, or combinations thereof, well known to one of ordinary skill in the art reading this disclosure.

[00018] Each of the client devices 14(1)-14(n) enables a user to request, obtain, and interact with one or more network resources, *e.g.*, web pages from one or more web sites, hosted by server devices 16(1)-16(n) through the proxy server device 12 via one or more communication networks (*e.g.*, communication network 18(1)), although one or more of the client devices 14(1)-14(n) could access

content and utilize other types and numbers of applications from other sources and could provide a wide variety of other functions for the user. Although multiple client devices 14(1)-14(n) are shown, other numbers and types of user computing systems could be used. In one example, the client devices 14(1)-14(n) comprise mobile devices with Internet access that permit a website form page or other retrieved data that is a part of a requested network resource to be displayed, although each of the client devices 14(1)-14(n) can connect to server devices 16(1)-16(n) via other types of network connections directly or indirectly, depending upon specific scenarios, as can be contemplated by one of ordinary skill in the art, after reading this disclosure. By way of example only, one or more of the client devices 14(1)-14(n) can comprise smart phones, personal digital assistants, computers, or other computing devices.

[00019] Each of client devices 14(1)-14(n) in this example is a computing device that includes a central processing unit (CPU) or processor 20, a memory 22, user input device 24, a display 26, and an interface system 28, and which are coupled together by a bus 30 or other link, although one or more of client devices 14(1)-14(n) can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 20 in each of client devices 14(1)-14(n) executes a program of stored instructions for aiding one or more aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions.

[00020] The memory 22 in each of the client devices 14(1)-14(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein as well as updated cookies associated with a network resource and received as part of one or more redirect chains forwarded by proxy server device 12, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor 20 can be used for the memory 22 in each of the client devices 14(1)-14(n).

[00021] The user input device 24 in each of the client devices 14(1)-14(n) is used to input selections, such as requests for a network resource, *e.g.*, a particular website form page or to enter data in fields of a form page, although the user input device could be used to input other types of data and interact with other elements of exemplary environment 10. The user input device 24 can include keypads, touch screens, and/or vocal input processing systems, although other types and numbers of user input devices can be used.

[00022] The display 26 in each of the client devices 14(1)-14(n) is used to show data and information to the user, such as website page by way of example only. The display in each of the client devices 14(1)-14(n) is a mobile phone screen display, although other types and numbers of displays could be used depending on the particular type of client device, as can be contemplated by one of ordinary skill in the art, after reading this disclosure.

[00023] The interface system 28 in each of the client devices 14(1)-14(n) is used to operatively couple and communicate between the client devices 14(1)-14(n) and the proxy server device 12 and server devices 16(1)-16(n) over the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00024] The server devices 16(1)-16(n) provide one or more pages from one or more web sites for use by one or more of the client devices 14(1)-14(n) via the proxy server device 12, although the server devices 16(1)-16(n) can provide other numbers and types of applications and/or content and can have provide other numbers and types of functions. Although server devices 16(1)-16(n) are shown for ease of illustration and discussion, other numbers and types of server systems, for example, web servers, and devices can be used. In one example, server devices 16(1)-16(n) can be web servers having dedicated hardware with software executing on the dedicated hardware to facilitate the proxy server device 12 and client devices 14(1)-14(n) in their functioning. In another example, server devices 16(1)-16(n) can be content servers that are configured to deliver network resources stored thereupon using the HTTP protocol, or other network protocols

for example. Content stored on server devices 16(1)-16(n) that can be part of the network resources requested by client devices 14(1)-14(n) can be web pages, electronic files and documents, configuration data, metadata, or other network data and files, by way of example only and not by way of limitation.

[00025] Each of the server devices 16(1)-16(n) include a central processing unit (CPU) or processor, a memory, and an interface system which are coupled together by a bus or other link, although each of the server devices 16(1)-16(n) could have other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor in each of the server devices 16(1)-16(n) executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions. When one of the server devices 16(1)-16(n) does not store the requested content, the server device may respond by sending a redirect message to the proxy server device 12, which the proxy server device 12 sends to the appropriate server device indicated by the redirect message instead of forwarding the redirect message back to the requesting one of the client devices 14(1)-14(n).

[00026] The memory in each of the server devices 16(1)-16(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated by way of the embodiments, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor, can be used for the memory in each of the server devices 16(1)-16(n).

[00027] The interface system in each of the server devices 16(1)-16(n) is used to operatively couple and communicate between the server devices 16(1)-16(n) and the proxy server device 12 and the client devices 14(1)-14(n) via

communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00028] Although embodiments of the proxy server device 12, the client devices 14(1)-14(n), and the server devices 16(1)-16(n), are described and illustrated herein, each of the client devices 14(1)-14(n), the proxy server device 12, and the server devices 16(1)-16(n), can be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s).

[00029] Furthermore, each of the systems of the embodiments may be conveniently implemented using one or more general purpose computer systems having computer readable medium, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[00030] In addition, two or more computing systems or devices can be substituted for any one of the systems in any embodiment of the embodiments. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the embodiments. The embodiments may also be implemented on computer system or systems that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (*e.g.*, voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[00031] The embodiments may also be embodied as a computer readable medium having instructions stored thereon for one or more aspects of the present invention as described and illustrated by way of the embodiments herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the embodiments, as described and illustrated herein.

[00032] Referring to FIG. 2, an exemplary redirect chain 200 is illustrated for a conventional scenario where proxy server device 12 is not configured to automatically handle redirect messages. It is to be noted that the sequence of steps for handling redirect chain 200 is only exemplary and one of ordinary skill in the art, after reading this disclosure, can contemplate alternative sequences of steps that achieve substantially the same result as shown by the sequence of steps for redirect chain 200.

[00033] More specifically, an example of redirect chain 200 including multiple redirects to obtain a network resource is illustrated in FIG. 2 starting at step 202 where one of the client devices 14(1)-14(n) (*e.g.*, a mobile device) sends an HTTP request for a network resource at a Uniform Resource Locator (URL) http://www.example.com/A.html, although other types of requests for other types of network resources may be sent. Although in this example one of the client devices 14(1)-14(n) via a web browser requests a page A.html at the website, "www.example.com" as shown, by way of example only, client devices 14(1)-14(n) may send a request for a network shared data file using a file transfer protocol instead of a URL using the HTTP protocol. In one example, this request is transmitted to the proxy server device 12 which processes and transmits the request to the one of the server device 16(1)-16(n) hosting the exemplary website www.example.com.

[00034] In step 204, the client device requesting the network resource gets a response from the hosting one of the server devices 16(1)-16(n) storing the network resource, in this example, the requested page A.html. The response includes a temporary redirect message 204a (shown as an exemplary status code 302 with a message "Moved Temporarily") to a different URL 204b. URL 204b

is shown exemplarily as http://www.example.com/B.html with a header field 204c shown as "Set-Cookie" including a cookie named "SESSION" set to value "1234," with domain equal to "example.com" and the path equal to "/."

[00035] Generally, the cookie is a string formed by the pair "name=value" (*e.g.*, "SESSION=1234", followed by optional attributes, like those in this example indicating the server domain(s) and path accepting this cookie. Although one illustrative example is described herein, this technology can be used with specifications for all cookies.

[00036] In step 206, the client device will then send a new HTTP request 206a to the server devices 16(1)-16(n) for a URL http://www.example.com/B.html also passing the cookie 206b shown as "SESSION=1234." In response, at step 208 of the redirect chain 200, the requesting client device obtains a new temporary redirect message 208b to another URL http://www.example.com/C.html with a new cookie named "LANG" set to a value "en" in a header field 208c.

[00037] In step 210, since the response at step 208 included a new cookie "LANG" with the new redirect response message 208b to a URL http://www.example.com/C.html, the client device will then send a new HTTP request 210a for http://www.example.com/C.html passing the two cookies "LANG=en" and "SESSION=1234" in a header 210b. In response, in step 212, the client device will get a real network resource 212b as a response from the hosting one of server devices 16(1)-16(n) with a status identifier message 212a shown as status code 200 set to "OK." It is to be noted although steps 202-212 are shown, a higher or a lower number of steps may be realized to obtain the real network resource 212b in response to the initial request in step 202 from one of the client devices 14(1)-14(n) for the network resource A.html by back and forth redirect communications as shown by redirect chain 200 between the client device 14(1)-14(n) and server devices 16(1)-16(n) via proxy server device 12. By way of example only, in some scenarios the client device may never be able to obtain the network resources for various reasons, including non-availability of the resources.

[00038] Referring to FIG. 3, an exemplary aspect of the present technology will now be described. FIG. 3 shows an exemplary redirect chain 300 when handled by proxy server device 12 interposed between client devices 14(1)-14(n) and server devices 16(1)-16(n). Advantageously, in this exemplary embodiment, back and forth communication between the client devices 14(1)-14(n) and hosting server devices 16(1)-16(n) with respect to intermediate redirect message portions of redirect chain 300 received for a requested network resource is reduced/minimized or even eliminated, thereby reducing latency in network connection between client devices 14(1)-14(n) and hosting server devices 16(1)-16(n). Redirect chain 300 is handled by proxy server device 12 as illustrated in the sequence of steps shown in FIG. 3 and described below, although the sequence of steps for handling redirect chain 300 is only exemplary and one of ordinary skill in the art, after reading this disclosure, can contemplate alternative sequences of steps that achieve the same, or substantially the same result as shown by the sequence of steps for redirect chain 300.

[00039] In step 302, one of client devices 14(1)-14(n) sends an initial request for a URI, *e.g.*, a URL http://www.example.com/A.html to the proxy server device 12. In step 304, unlike steps 204-208 of FIG. 2, proxy server device 12 executes all the intermediate redirects on its side and returns a temporary redirect response message 304a (shown as an exemplary status code 302 with a message "Moved Temporarily") for a URL 304b identified as a URL http://www.example.com/C.html to the requesting one of client devices 14(1)-14(n). In this example, both "LANG" and "SESSION" cookies will be set as shown by respective "Set-Cookie" header fields 304c such that they correspond to the updated state of the requested URL www.example.com. Although a single redirect message is illustrated in FIG. 3, the process can be extended to proxy server device 12 handling all the redirect messages from server devices 14(1)-14(n) until a real resource (*e.g.*, with an HTTP status message 200) is obtained. The information, including cookies, contained in the intermediate or temporary redirect response message 304a is forwarded to the requesting one of client devices 14(1)-14(n). For example, the forwarded message can include merged cookies previously received.

[00040] In step 306, the requesting client device will then send a request including a last redirect message 306a from the redirect chain 300 to the proxy server device 12 for the network resource URL http://www.example.com/C.html with updated cookies "SESSION" and "LANG" in the header field 306b as illustrated. The requesting client device forms the last redirect message 306a based upon the information forwarded by proxy server device 12 in step 304, and cookie information stored at the requesting client device because of prior navigation by the requesting client device, although the requesting client device may obtain the updated state of cookies from other sources, for example, one of server devices 16(1)-16(n).

[00041] In step 308, in response to the request sent in step 306, the requesting one of the client devices 14(1)-14(n) gets a real network resource 308b as a response from the hosting one of the server devices 16(1)-16(n) with a status identifier 308a set to "OK" (shown as status code 200). Advantageously, the redirect chain 300 is executed substantially between the proxy server device 12 and the hosting one of server devices 16(1)-16(n) in a substantially lesser time as compared to the intermediate redirect responses shown in redirect chain 200 exchanged substantially between the client devices 14(1)-14(n) and the hosting one of server devices 16(1)-16(n) with minimal to zero functionality of proxy server device 12 with respect to handling redirect response message chain 200. This is because typically there is a faster connection between proxy server device 12 and server devices 16(1)-16(n) than between client devices 14(1)-14(n) and server devices 16(1)-16(n).

[00042] Referring to FIG. 4, an exemplary method for reducing latency in network connections (*e.g.*, HTTP connections) will now be described using a flowchart 400 with reference back to FIGS. 1-3. In this example, an exemplary sequence of steps performed by proxy server device 12 to handle redirects as discussed above is described. The flowchart 300 is representative of example machine readable instructions to implementing reducing latency in network connections, for example, at proxy server device 12. In this example, the machine readable instructions comprise an algorithm for execution by: (a) a processor (*e.g.*, CPU 13), (b) a controller, and/or (c) one or more other suitable processing

device(s) within proxy server device 12, for example. The algorithm may be implemented in software stored on tangible computer readable media such as, for example, a flash memory, a CD-ROM, a floppy disk, a hard drive, a digital video (versatile) disk (DVD), or other memory devices, but persons of ordinary skill in the art will readily appreciate that the entire algorithm and/or parts thereof could alternatively be executed by a device other than a processor and/or implemented in firmware or dedicated hardware in a well known manner (*e.g.*, it may be implemented by an application specific integrated circuit (ASIC), a programmable logic device (PLD), a field programmable logic device (FPLD), a field programmable gate array (FPGA), discrete logic, or the like). For example, at least some of the components of the proxy server device 12 could be implemented by software, hardware, and/or firmware. Also, some or all of the machine readable instructions represented by the process of flowchart 400 of FIG. 4 may be implemented manually at the proxy server device 12, for example, using a command line interface (CLI) prompt window operated by a system administrator. Further, although the example algorithm is described with reference to flowchart 400, persons of ordinary skill in the art will readily appreciate that many other methods of implementing the example machine readable instructions may alternatively be used. For example, the order of execution of the blocks in flowchart 400 may be changed, and/or some of the blocks described may be changed, eliminated, or combined.

[00043] The flow begins in step 402, when proxy server device 12 receives a request from one of the client devices 14(1)-14(n). By way of example only, the request can be an HTTP request, although other types of requests (*e.g.*, any type of HxxP or HTTPS request). In step 406, the received request is processed by proxy server device 12 and forwarded to one of the content hosting server devices 16(1)-16(n) based upon the information in the request. In response, in step 408, proxy server device 12 receives a first response from the hosting one of the server devices 16(1)-16(n), the first response being formed based upon the information in the forwarded request received by the hosting one of server devices 16(1)-16(n). In step 410, proxy server device 12 determines whether the response from the hosting one of server devices 16(1)-16(n) is a real network resource (*e.g.*, a status

code 200 message) or a redirect message. If a real resource is received, the Yes branch is followed to step 412, where proxy server device 12 sends the first response including the link to the actual network resource to the requesting client device.

[00044] However, as shown in step 414, if the first response from the hosting one of the server devices 16(1)-16(n) is not a real network resource but is a redirect message (for example, a temporary redirect 204a with status code 302, shown in FIG. 2), proxy server device 12 will collect the cookies set in the first response and merge them with those sent with the last and/or prior requests for the network resource, including the original cookie received by the proxy server device 12 from the hosting one of the server devices 16(1)-16(n).

[00045] In step 416, proxy server device 12 will then send a new request (or, a second request) for the network resource (*e.g.*, URL "www.example.com/C.html") specified in the last redirect response (*e.g.*, redirect message 306a) to the hosting one of the server devices 16(1)-16(n), passing the cookies used in the first request (shown in step 302 of FIG. 3) and all the cookies collected from the previous redirect response messages 304a, 306a. It is to be noted the proxy server device 12 does not send the redirect messages that do not include real network resource to the client devices 14(1)-14(n). In step 418, proxy server device 12 receives a response back from the hosting one of server devices 16(1)-16(n), which response is examined again, similar to step 410. In step 420, if the response from the hosting one of server devices 16(1)-16(n) is again a redirect message (*e.g.*, redirect message 306a), then the proxy server device 12 will continue to repeat steps 414-418 by subsequently creating a new request for the server devices 16(1)-16(n) based upon the last redirect response message obtained from server devices 16(1)-16(n). Using the technique discussed above, instead of the requesting one of client devices 14(1)-14(n) handling all intermediate redirect messages from server devices 14(1)-104(n), proxy server device 12 is configured to handle the intermediate redirects until a real network resource is obtained from server devices 16(1)-16(n). An advantage of the proxy server device 12 handling the redirect message chain instead of client devices 14(1)-14(n) is to reduce latency in provisioning requests from client devices 14(1)-14(n) since connection

between proxy server device 12 and server devices 16(1)-16(n) is faster than connection between client devices 14(1)-14(n) and server devices 16(1)-16(n).

[00046] However, if the response from the hosting one of server devices 16(1)-16(n) is a real network resource, in step 422, proxy server device 12 will send a redirect response 304a to the requesting one of the client devices 14(1)-14(n) specifying the last URL 304b of the redirect chain 300 and all the cookies (*e.g.*, cookies in header field 306b) collected from the redirect response messages 304a, 306a. Subsequently, the requesting one of the client devices 14(1)-14(n) can use the most updated URL 304b to the hosting one of the server devices 16(1)-16(n) via proxy server device 12, and receive the requested network resource. It is to be noted although in the examples above, URLs are being discussed, the technology works in substantially the same manner for other types of resource identifier schemes, *e.g.*, Uniform Resource Names (URNs), as URLs are being discussed by way of example only and not by way of limitation, as can be contemplated by one of ordinary skill in the art after reading this disclosure. Accordingly, the steps shown in flowchart 400 advantageously reduce the slow communication of intermediate redirect response messages between the client devices 14(1)-14(n) and the hosting server devices 16(1)-16(n), thus reducing latency in the network connection, which in this example is an HTTP connection, although other types of connections (*e.g.*, FTP connections) may be used.

[00047] Accordingly, as illustrated and described herein this technology provides a number of advantages including providing a method, computer readable medium and an apparatus that reduces latency in network connections, for example, by managing handling of redirect messages from server devices 16(1)-16(n) substantially at proxy server device 12 interposed between the client devices 14(1)-14(n) and server devices 16(1)-16(n) on which various network resources are stored, which network resources are requested by client devices 14(1)-14(n). With this technology, the proxy server device 12 optimizes network resource provisioning by returning only the last redirect message along with all cookies collected from intermediate redirect messages from a chain of redirect messages from server devices 16(1)-16(n) to client devices 14(1)-14(n) for a

particular network resource, thereby reducing latency in network connections between client devices and server devices.

[00048] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for reducing latency in network connections, the method comprising:

receiving at a proxy server device a first response from one or more server devices in response to a first request from a client device for a network resource stored on the one or more server devices;

determining at the proxy server device whether or not the first response is a redirect message; and

sending a second request from the proxy server device to the one or more server devices when the first response includes the first redirect message, wherein the sending is performed at least partially based upon the received redirect message.

2. The method as set forth in claim 1, wherein the proxy server device is configured to collect one or more cookies received in the first response, and wherein the sending the second request comprises:

sending the collected one or more cookies merged with one or more cookies received in one or more prior responses to the one or more server devices as a part of the second request.

3. The method as set forth in claim 2, wherein the collected one or more cookies merged with the one or more cookies received in the one or more prior responses are forwarded to the client device by the proxy server device.

4. The method as set forth in claim 1 further comprising:

receiving a second response from the one or more server devices based upon the sending the second request; and

providing the second response to the client device when the second response includes a network link to the network resource.

5. The method as set forth in claim 4, wherein the providing comprises:

sending to the client device from the proxy server device a last changed network address associated with the network resource and updated cookies to the client device for obtaining the network resource in a future request for the network resource from the client device.

6. The method as set forth in claim 4, wherein the network link is a Uniform Resource Identifier (URI).

7. The method as set forth in claim 1, wherein the sending is performed until a network link to a real network resource is obtained from the one or more server devices.

8. The method as set forth in claim 1, wherein the first redirect message and subsequent redirect messages that do not include a real network resource are not sent to the client device until a network link to a real network resource is obtained from the one or more server devices.

9. The method as set forth in claim 1, wherein the client device is a mobile client device.

10. A computer readable medium having stored thereon instructions for reducing latency in network connections comprises machine executable code which when executed by at least one processor, causes the processor to perform steps including:

receiving at a proxy server device a first response from one or more server devices in response to a first request from a client device for a network resource stored on the one or more server devices;

determining at the proxy server device whether or not the first response is a redirect message; and

sending a second request from the proxy server device to the one or more server devices when the first response includes the first redirect

message, wherein the sending is performed at least partially based upon the received redirect message.

11. The medium as set forth in claim 10, wherein the proxy server device is configured to collect one or more cookies received in the first response, and wherein the sending the second request comprises:

sending the collected one or more cookies merged with one or more cookies received in one or more prior responses to the one or more server devices as a part of the second request.

12. The medium as set forth in claim 11, wherein the collected one or more cookies merged with the one or more cookies received in the one or more prior responses are forwarded to the client device by the proxy server device.

13. The medium as set forth in claim 10, wherein the machine executable code which when executed by at least one processor, causes the processor to further perform steps including:

receiving a second response from the one or more server devices based upon the sending the second request; and

providing the second response to the client device when the second response includes a network link to the network resource.

14. The medium as set forth in claim 13, wherein the providing comprises:

sending to the client device from the proxy server device a last changed network address associated with the network resource and updated cookies to the client device for obtaining the network resource in a future request for the network resource from the client device.

15. The medium as set forth in claim 13, wherein the network link is a Uniform Resource Identifier (URI).

16. The medium as set forth in claim 10, wherein the sending is performed until a network link to a real network resource is obtained from the one or more server devices.

17. The medium as set forth in claim 10, wherein the first redirect message and subsequent redirect messages that do not include a real network resource are not sent to the client device until a network link to a real network resource is obtained from the one or more server devices.

18. The medium as set forth in claim 10, wherein the client device is a mobile client device.

19. An apparatus configured to reduce latency in network connections comprising one or more processors and a memory coupled to the one or more processors configured to execute programmed instructions stored in the memory including instructions for implementing:

receiving at a proxy server device a first response from one or more server devices in response to a first request from a client device for a network resource stored on the one or more server devices;

determining at the proxy server device whether or not the first response is a redirect message; and

sending a second request from the proxy server device to the one or more server devices when the first response includes the first redirect message, wherein the sending is performed at least partially based upon the received redirect message.

20. The apparatus as set forth in claim 19, wherein the proxy server device is configured to collect one or more cookies received in the first response, and wherein the sending the second request comprises:

sending the collected one or more cookies merged with one or more cookies received in one or more prior responses to the one or more server devices as a part of the second request.

21. The apparatus as set forth in claim 20, wherein the collected one or more cookies merged with the one or more cookies received in the one or more prior responses are forwarded to the client device by the proxy server device.

22. The apparatus as set forth in claim 20, wherein the programmed instructions further include instructions for implementing:

receiving a second response from the one or more server devices based upon the sending the second request; and

providing the second response to the client device when the second response includes a network link to the network resource.

23. The apparatus as set forth in claim 22, wherein the providing comprises:

sending to the client device from the proxy server device a last changed network address associated with the network resource and updated cookies to the client device for obtaining the network resource in a future request for the network resource from the client device.

24. The apparatus as set forth in claim 22, wherein the network link is a Uniform Resource Identifier (URI).

25. The apparatus as set forth in claim 19, wherein the sending is performed until a network link to a real network resource is obtained from the one or more server devices.

26. The apparatus as set forth in claim 19, wherein the first redirect message and subsequent redirect messages that do not include a real network resource are not sent to the client device until a network link to a real network resource is obtained from the one or more server devices.

27. The apparatus as set forth in claim 19, wherein the client device is a mobile client device.

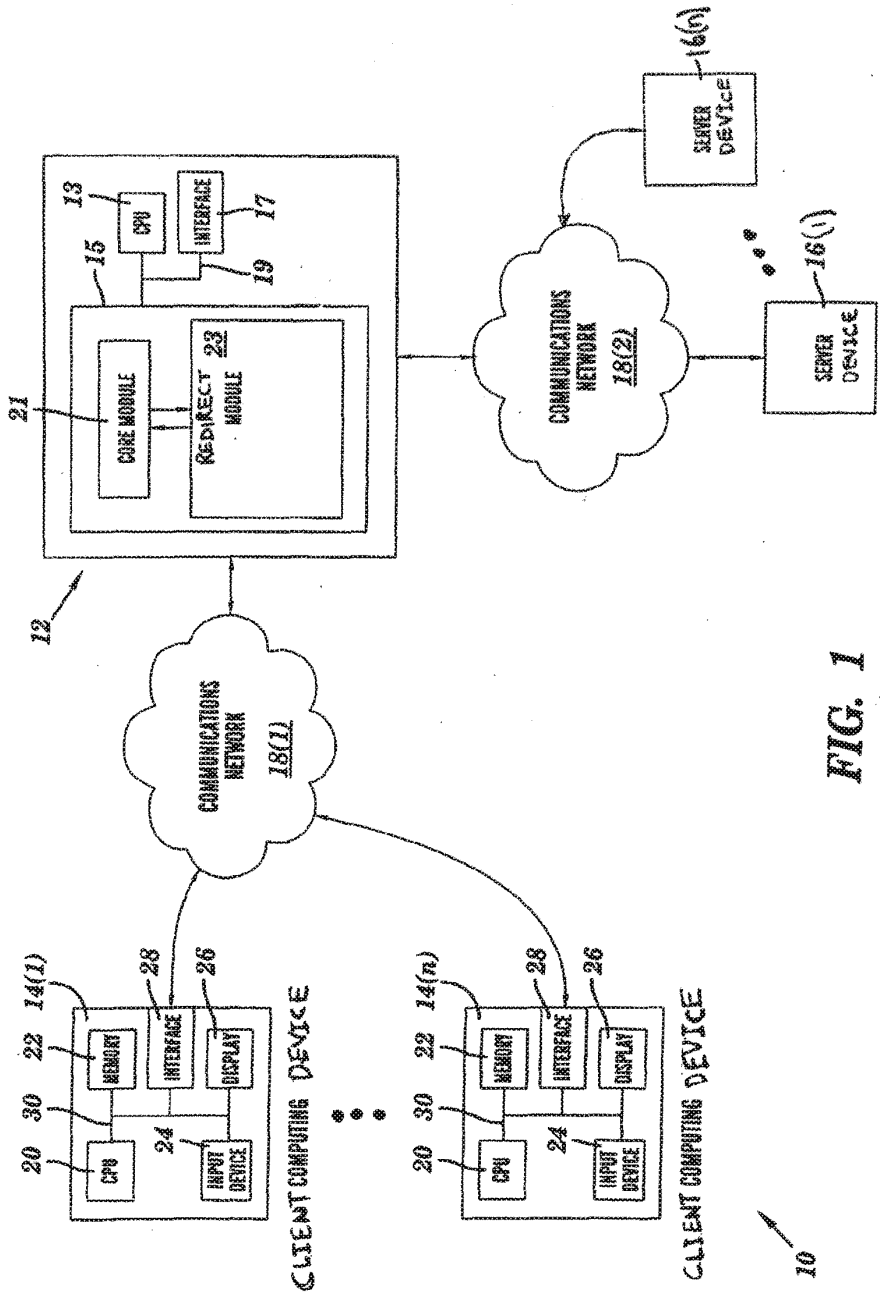

SERVER DEVICE 16(n)
SERVER DEVICE 16(1)
COMMUNICATIONS NETWORK 18(2)
COMMUNICATIONS NETWORK 18(1)
CORE MODULE
REDIRECT MODULE 23
CPU
INTERFACE
MEMORY
DISPLAY
INPUT DEVICE
CLIENT COMPUTING DEVICE
CLIENT COMPUTING DEVICE
10
12
13
15
17
19
20
21
22
24
26
28
30
14(1)
14(n)

FIG. 1

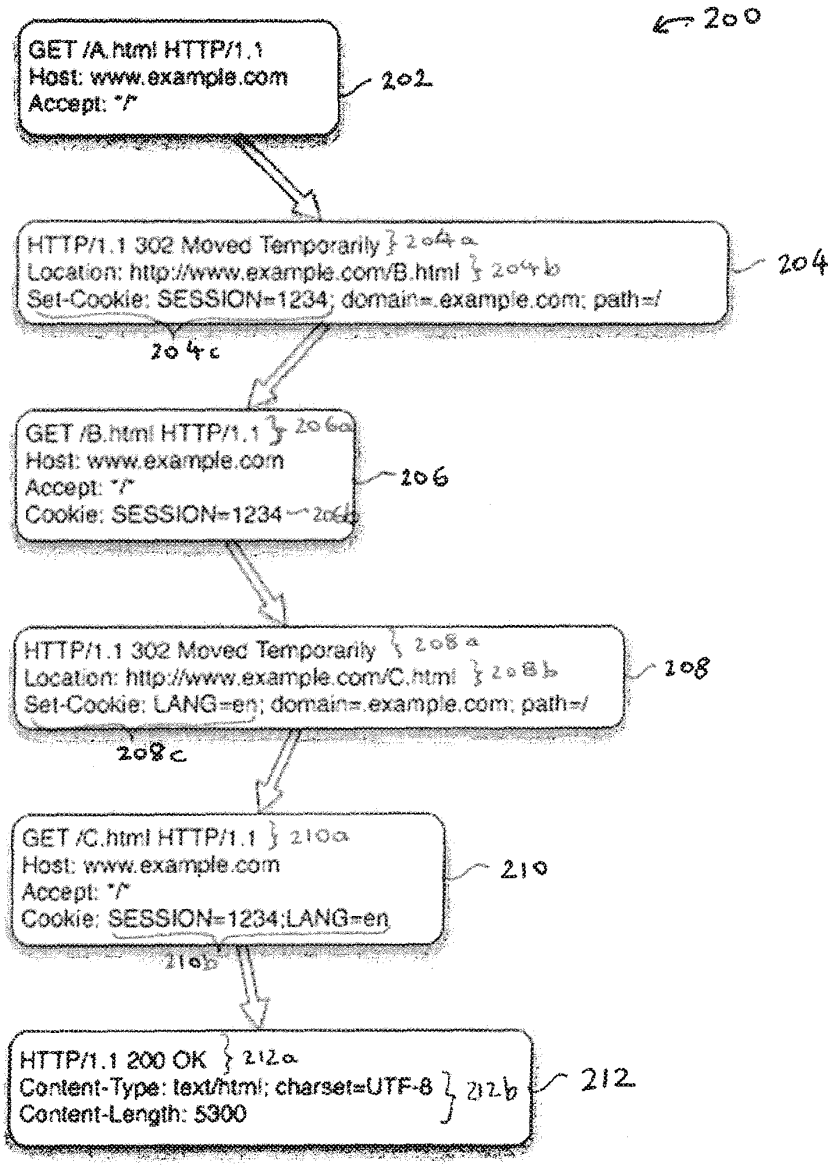

200
GET /A.html HTTP/1.1
Host: www.example.com
Accept: */*
202
HTTP/1.1 302 Moved Temporarily
Location: http://www.example.com/B.html
Set-Cookie: SESSION=1234; domain=.example.com; path=/
204
204a
204b
204c
GET /B.html HTTP/1.1
Host: www.example.com
Accept: */*
Cookie: SESSION=1234
206
206a
206b
HTTP/1.1 302 Moved Temporarily
Location: http://www.example.com/C.html
Set-Cookie: LANG=en; domain=.example.com; path=/
208
208a
208b
208c
GET /C.html HTTP/1.1
Host: www.example.com
Accept: */*
Cookie: SESSION=1234;LANG=en
210
210a
210b
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Content-Length: 5300
212
212a
212b

FIG. 2

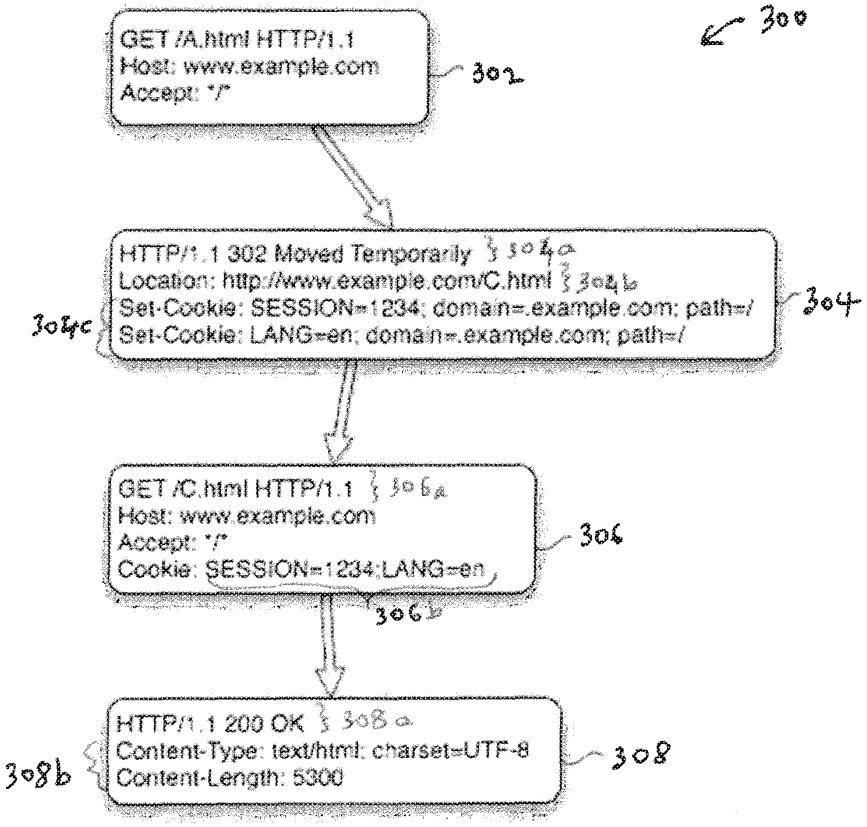

300
GET /A.html HTTP/1.1
Host: www.example.com
Accept: */*
302
HTTP/1.1 302 Moved Temporarily
Location: http://www.example.com/C.html
Set-Cookie: SESSION=1234; domain=.example.com; path=/
Set-Cookie: LANG=en; domain=.example.com; path=/
304
304a
304b
304c
GET /C.html HTTP/1.1
Host: www.example.com
Accept: */*
Cookie: SESSION=1234;LANG=en
306
306a
306b
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Content-Length: 5300
308
308a
308b

FIG. 3

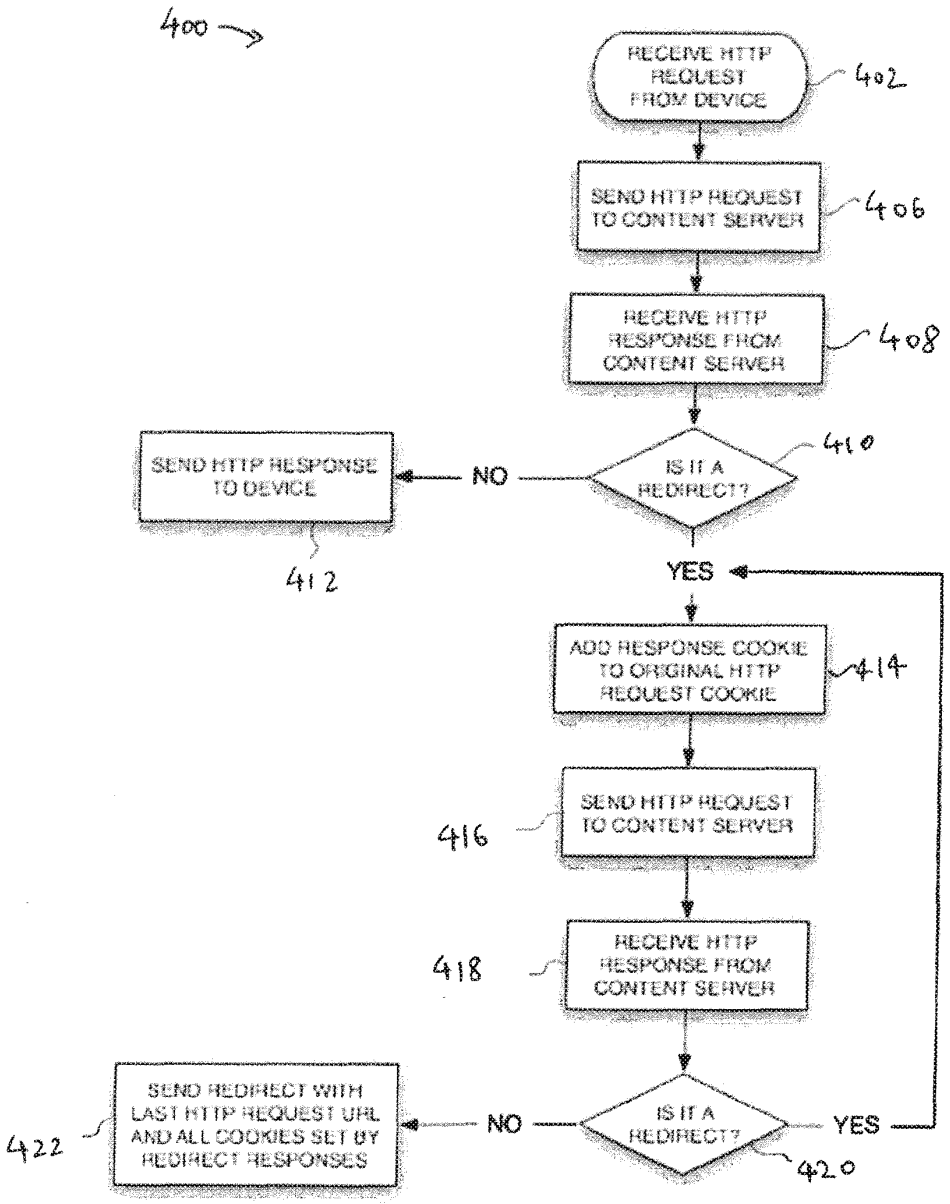

400
RECEIVE HTTP REQUEST FROM DEVICE
402
SEND HTTP REQUEST TO CONTENT SERVER
406
RECEIVE HTTP RESPONSE FROM CONTENT SERVER
408
IS IT A REDIRECT?
410
NO
SEND HTTP RESPONSE TO DEVICE
412
YES
ADD RESPONSE COOKIE TO ORIGINAL HTTP REQUEST COOKIE
414
SEND HTTP REQUEST TO CONTENT SERVER
416
RECEIVE HTTP RESPONSE FROM CONTENT SERVER
418
IS IT A REDIRECT?
420
NO
YES
SEND REDIRECT WITH LAST HTTP REQUEST URL AND ALL COOKIES SET BY REDIRECT RESPONSES
422

FIG. 4

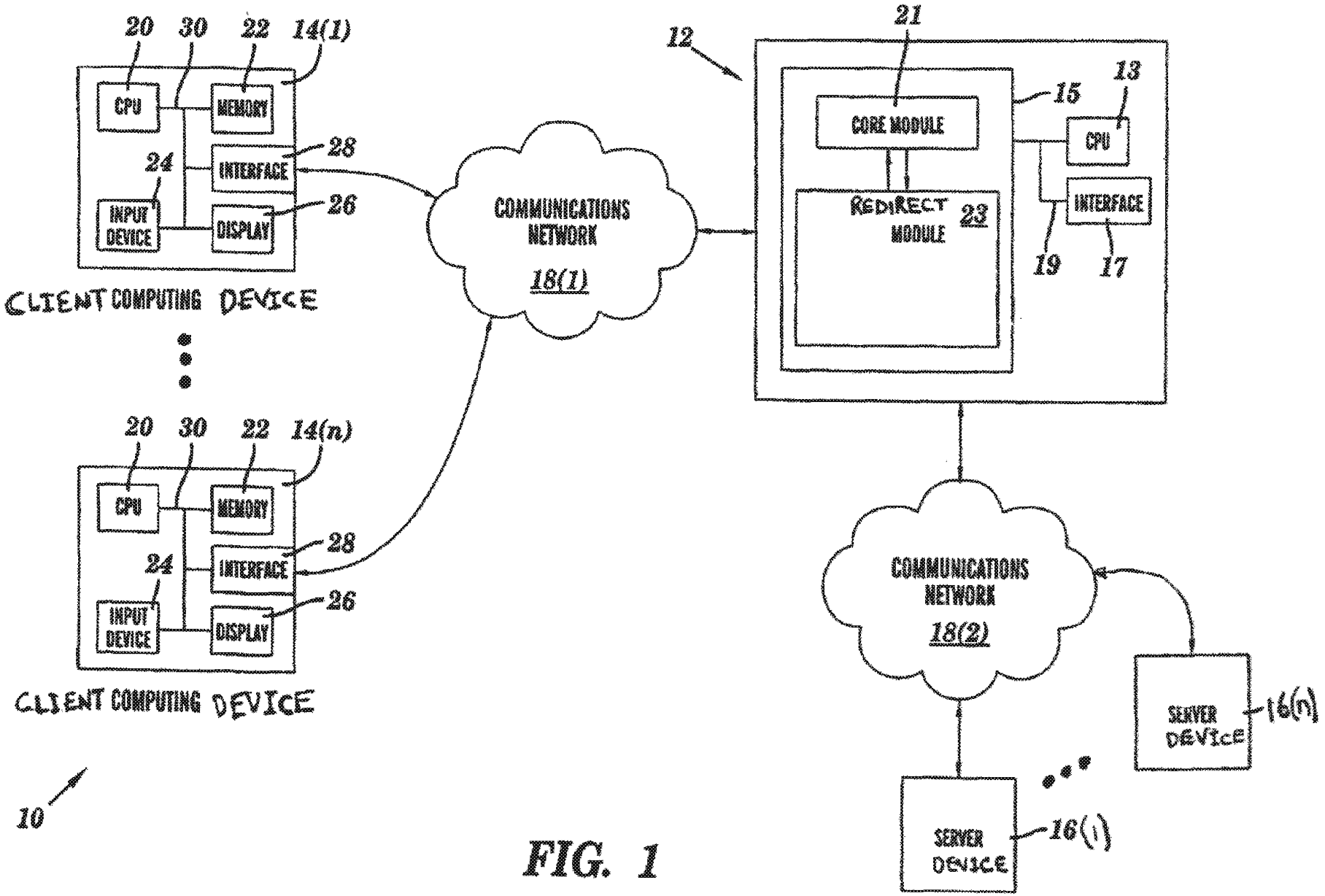


FIG. 1

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2816338 A1 2012/05/18

(21) 2 816 338

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(86) Date de dépôt PCT/PCT Filing Date: 2011/11/09
(87) Date publication PCT/PCT Publication Date: 2012/05/18
(85) Entrée phase nationale/National Entry: 2013/04/26
(86) N° demande PCT/PCT Application No.: US 2011/059994
(87) N° publication PCT/PCT Publication No.: 2012/064857
(30) Priorités/Priorities: 2010/11/09 (US12/927,169); 2011/07/13 (US13/135,707)

(51) Cl.Int./Int.Cl. *H04L 29/06* (2006.01), *G06F 15/16* (2006.01)
(71) Demandeur/Applicant:
USABLENET INC., US
(72) Inventeur/Inventor:
SCODA, ENRICO, IT
(74) Agent: GOWLING LAFLEUR HENDERSON LLP

(54) Titre : PROCEDES DE REDUCTION DE LATENCE DANS DES CONNEXIONS DE RESEAU A L'AIDE DE REDIRECTIONS AUTOMATIQUES, ET SYSTEMES CORRESPONDANTS

(54) Title: METHODS FOR REDUCING LATENCY IN NETWORK CONNECTIONS USING AUTOMATIC REDIRECTS AND SYSTEMS THEREOF

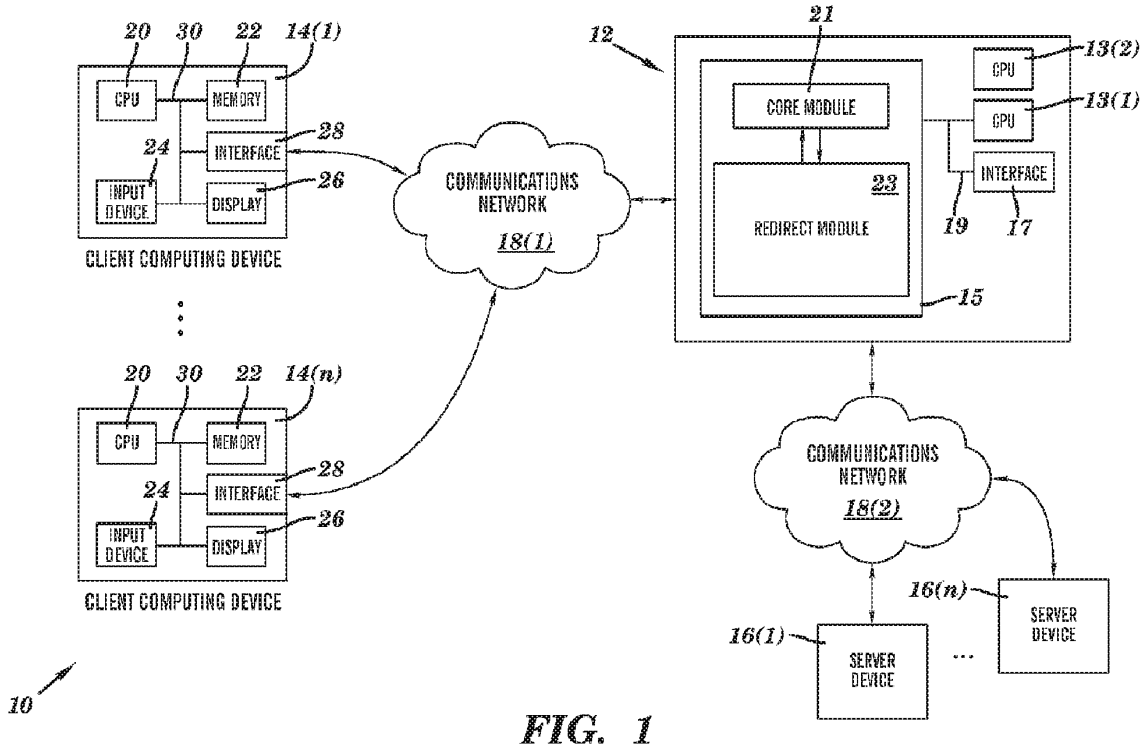


FIG. 1

(57) Abrégé/Abstract:
A method, computer readable medium, and apparatus for reducing latency in network connections include receiving at a web content optimization apparatus a request from one of one or more client devices for network content hosted at one of one or more

(21) 2 816 338

(13) A1

(57) Abrégé(suite)/Abstract(continued):
server devices. The web content optimization apparatus determines whether an automatic redirect parameter is enabled for the request. The request from one proxy processing device acting as the requesting one of the client devices within the web content optimization apparatus is sent to another proxy processing device within the web content optimization apparatus when the automatic redirect parameter is determined to be enabled. One or more redirect messages with one or more optimized cookies associated with the requested network resource are obtained from the one or more server devices at the another proxy processing device within the web content optimization apparatus. The last of the one or more redirects messages with the one or more optimized cookies when the requested network resource is located is sent from the another proxy processing device within the web content optimization apparatus to the one proxy processing device within the web content optimization apparatus. The last of the one or more redirect messages with the one or more optimized cookies associated with the requested network resource is sent from the web content optimization apparatus to the requesting one of the one or more of client devices.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
18 May 2012 (18.05.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/064857 A3

(51) International Patent Classification:
H04L 29/06 (2006.01) *G06F 15/16* (2006.01)

(21) International Application Number:
PCT/US2011/059994

(22) International Filing Date:
9 November 2011 (09.11.2011)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/927,169 9 November 2010 (09.11.2010) US
13/135,707 13 July 2011 (13.07.2011) US

(71) Applicant *(for all designated States except US)*: **US-ABLENET INC.** [US/US]; 28 W. 23rd Street, 6th Floor, New York, NY 10010 (US).

(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: **SCODA, Enrico** [IT/IT]; Via Cividina 416/3, Martignacco(UD) (IT).

(74) Agents: **LEINBERG, Gunnar, G.** et al.; LeClairRyan, A Professional Corporation, 70 Linden Oaks, Suite 210, Rochester, NY 14625 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

(88) Date of publication of the international search report:
27 September 2012

(54) Title: METHODS FOR RUDUCING LATENCY IN NETWORK CONNECTIONS USING AUTOMATIC REDIRECTS AND SYSTEMS THEREOF

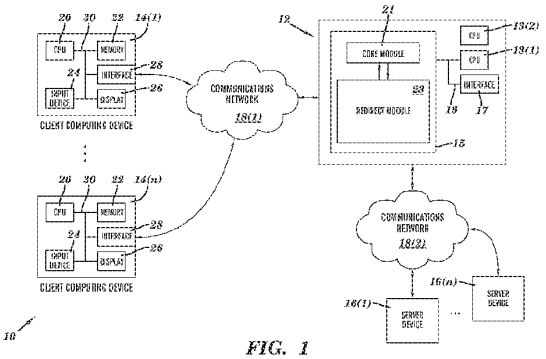


FIG. 1

(57) Abstract: A method, computer readable medium, and apparatus for reducing latency in network connections include receiving at a web content optimization apparatus a request from one of one or more client devices for network content hosted at one of one or more server devices. The web content optimization apparatus determines whether an automatic redirect parameter is enabled for the request. The request from one proxy processing device acting as the requesting one of the client devices within the web content optimization apparatus is sent to another proxy processing device within the web content optimization apparatus when the automatic redirect parameter is determined to be enabled. One or more redirect messages with one or more optimized cookies associated with the requested network resource are obtained from the one or more server devices at the another proxy processing device within the web content optimization apparatus. The last of the one or more redirects messages with the one or more optimized cookies when the requested network resource is located is sent from the another proxy processing device within the web content optimization apparatus to the one proxy processing device within the web content optimization apparatus. The last of the one or more redirect messages with the one or more optimized cookies associated with the requested network resource is sent from the web content optimization apparatus to the requesting one of the one or more of client devices.

WO 2012/064857 A3

METHODS FOR REDUCING LATENCY IN NETWORK CONNECTIONS USING AUTOMATIC REDIRECTS AND SYSTEMS THEREOF

[0001] This application is a continuation-in-part of prior Application No. 12/927,169, filed November 9, 2010, which is herein incorporated by reference in its entirety.

FIELD

[0002] This technology generally relates to web content optimization apparatuses and, more particularly, to methods and systems for reducing latency in network connections using automatic redirects in web content optimization apparatuses.

BACKGROUND

[0003] When a client device connects to a server (*e.g.*, a web server, or a content server) to get a network resource using a network protocol, *e.g.*, the Hyper-text Transfer protocol (HTTP), the server responds by sending the network resource or by sending a redirect message back to the client device over a communication channel. If the client device receives a redirect message, it will need to send a new request to the server based upon the redirect message, and the server will again respond with a redirect or a real resource. This communication process between the client device and the server repeats until the client device is able to get the resource, if available.

[0004] However, when the client device, *e.g.*, a cell phone using a radio network, or a computer having a slow Internet connection, requests a network resource and has to perform more than one redirects to obtain the network resource, the client device will experience substantial delay and will spend a considerable amount of time to execute the whole process before finally being provided with the network resource. The delay can occur, for example, because of a large time to establish a connection and send the HTTP request, also referred to as latency time of radio networks or other slow network connections (*e.g.*, Internet

via a dial-up connection). Unfortunately, this delay can often lead to the client device not being able to obtain the network resource at all, or the client device giving up or relinquishing attempts to obtain the network resource under time constraints.

[0005] One conventional solution built to obtain faster HTTP responses for slow connection networks utilizes one or more proxy server devices (*e.g.*, web proxy servers). Another conventional solution utilizes telephone carrier data centers that handle the network traffic for each client device by handling one or more requests when the client device is a mobile telephone, or a mobile personal digital assistant (PDA) device, for example. Unfortunately, the above-noted conventional solutions do not resolve the redirection problem associated with network resources stored on the servers since the redirect messages are forwarded on to the client devices for handling resulting in multiple back and forth communication between the client devices and the servers.

SUMMARY

[0006] A method for reducing latency in network connections includes receiving at an optimization apparatus a request from one of one or more client devices for network content hosted at one of one or more server devices. The optimization apparatus determines whether an automatic redirect parameter is enabled for the request. The request from one proxy processing device acting as the requesting one of the client devices within the optimization apparatus is sent to another proxy processing device within the optimization apparatus when the automatic redirect parameter is determined to be enabled. One or more redirect messages with one or more optimized cookies associated with the requested network resource are obtained from the one or more server devices at the other proxy processing device within the web content optimization apparatus. The last of the one or more redirects messages with the one or more optimized cookies when the requested network resource is located is sent from the other proxy processing device within the optimization apparatus to the one proxy processing device within the optimization apparatus. The last of the one or more redirect messages with the one or more optimized cookies associated with the requested

network resource is sent from the optimization apparatus to the requesting one of the one or more of client devices.

[0007] A non-transitory computer readable medium having stored thereon instructions for reducing latency in network connections comprises machine executable code which when executed by at least one processor, causes the processor to perform steps including receiving a request from one of one or more client devices for network content hosted at one of one or more server devices. A determination is made whether an automatic redirect parameter is enabled for the request. The request from one proxy processing device acting as the requesting one of the client devices within an optimization apparatus is sent to another proxy processing device within the optimization apparatus when the automatic redirect parameter is determined to be enabled. One or more redirect messages with one or more optimized cookies associated with the requested network resource is obtained from the one or more server devices at the other proxy processing device within the optimization apparatus. The last of the one or more redirects messages with the one or more optimized cookies when the requested network resource is located is sent from the other proxy processing device within the optimization apparatus to the one proxy processing device within the optimization apparatus. The last of the one or more redirect messages with the one or more optimized cookies associated with the requested network resource is sent to the requesting one of the one or more of client devices.

[0008] An apparatus configured to reduce latency in network connections comprising one or more processors and a memory coupled to the one or more processors configured to execute programmed instructions stored in the memory including receiving a request from one of one or more client devices for network content hosted at one of one or more server devices. A determination is made whether an automatic redirect parameter is enabled for the request. The request from one proxy processing device acting as the requesting one of the client devices within an optimization apparatus is sent to another proxy processing device within the optimization apparatus when the automatic redirect parameter is determined to be enabled. One or more redirect messages with one or more optimized cookies associated with the requested network resource is obtained

from the one or more server devices at the other proxy processing device within the optimization apparatus. The last of the one or more redirects messages with the one or more optimized cookies when the requested network resource is located is sent from the other proxy processing device within the optimization apparatus to the one proxy processing device within the optimization apparatus. The last of the one or more redirect messages with the one or more optimized cookies associated with the requested network resource is sent to the requesting one of the one or more of client devices.

[0009] This technology provides a number of advantages including providing a method, computer readable medium and apparatus that efficiently manages the whole redirect chain on behalf of the client device by optimizing at least a portion of redirect messages automatically within a web content optimization apparatus. Since at least a portion of the redirect chain associated with locating and obtaining the network resource is handled internally by separate processing devices within the web content optimization apparatus, network communications between the client devices and the servers for network resources are sped up. This technology exploits the faster internal processing of redirect messages in one or more redirect chains by the web content optimization apparatus, as compared to slower communication speeds between client device and the servers.

BRIEF DESCRIPTION OF THE DRAWINGS

[00010] FIG. 1 is a block diagram of an exemplary network environment with a web content optimization apparatus interposed between client devices and server devices;

[00011] FIG. 2 is a flow chart of a method for processing an exemplary sequence of HTTP requests and responses between one of the client devices and the server devices hosting example.com and sample.com;

[00012] FIG. 3 is a flow chart of a method for processing an exemplary sequence of HTTP requests and responses between a client device and the web content optimization apparatus; and

[00013] FIG. 4 is a flow chart of a method for processing an exemplary sequence of HTTP requests and responses between a client device and the web content optimization apparatus when auto redirect is enabled; and

[00014] FIG. 5 is a flowchart of an exemplary method for reducing latency in network connections by optimizing network resource using automatic redirection of redirect response messages at the web content optimization apparatus.

DETAILED DESCRIPTION

[00015] An exemplary environment 10 in which a web content optimization apparatus 12 is optimized for reducing latency in network connections is illustrated in FIG. 1. The exemplary environment 10 includes the web content optimization apparatus 12, client devices 14(1)-14(n), server devices 16(1)-16(n), and communication networks 18(1)-18(2), although other numbers and types of systems, devices, and/or elements in other configurations and environments with other communication network topologies can be used. This technology provides a number of advantages including providing a method, computer readable medium, and an apparatus that reduces latency in network connections, for example, HTTP connections.

[00016] Referring more specifically to FIG. 1, the web content optimization apparatus 12 manages handling of redirect messages or redirection responses from the server devices 16(1)-16(n) for and/or on behalf of requesting client devices 14(1)-14(n) and provides updated cookie information to the client devices 14(1)-14(n) for future requests for network resources from the client devices 14(1)-14(n), although the web content optimization apparatus 12 can provide other numbers and types of functions and other types of processing devices can be used. Although one web content optimization apparatus 12 is shown, other numbers and types of optimization devices and systems can be used.

[00017] The web content optimization apparatus 12 includes central processing units (CPU) or processors 13(1) and 13(2), a memory 15, and an interface system 17 which are coupled together by a bus 19 or other link, although

other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processors 13(1) and 13(2) in the web content optimization apparatus 12 executes a program of stored instructions to carry out or perform one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although other numbers of processors can be used and one or more of the processors could execute other numbers and types of programmed instructions.

[00018] The memory 15 in the web content optimization apparatus 12 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processors 13(1) and 13(2), can be used for the memory 15 in the web content optimization apparatus 12. In these embodiments, the memory 15 includes a core module 21 and a redirect module 23 which store programmed instructions for one or more aspects of the present invention as described and illustrated herein, although the memory can comprise other types and numbers of systems, devices, and elements in other configurations which store other data.

[00019] The interface system 17 in the web content optimization apparatus 12 is used to operatively couple and communicate between the web content optimization apparatus 12 and the client devices 14(1)-14(n) and the server devices 16(1)-16(n) via the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used. By way of example only, the communication networks 18(1) and 18(2) can use TCP/IP over Ethernet and industry-standard protocols, including HTTP, HTTPS, WAP, and SOAP, although other types and numbers of communication networks, such as a direct connection, a local area network, a wide area network, modems and phone lines, e-mail, and wireless and hardwire communication technology, each having their own

communications protocols, can be used. In one exemplary embodiment, one of communication networks 18(1) and 18(2) can be operating over one or more low-speed connections (*e.g.*, a dial-up connection) while the other one of the communication networks 18(1) and 18(2) can be operating over a high speed, high bandwidth connection (*e.g.*, optical fiber based communication network). In yet another exemplary embodiment, one or more of communication networks 18(1) and 18(2) can be a radio network, a satellite network, an Internet connection, a wired cable network, or combinations thereof, well known to one of ordinary skill in the art reading this disclosure.

[00020] Each of the client devices 14(1)-14(n) enables a user to request, obtain, and interact with one or more network resources, *e.g.*, web pages from one or more web sites, hosted by server devices 16(1)-16(n) through the web content optimization apparatus 12 via one or more communication networks (*e.g.*, communication network 18(1)), although one or more of the client devices 14(1)-14(n) could access content and utilize other types and numbers of applications from other sources and could provide a wide variety of other functions for the user. Although multiple client devices 14(1)-14(n) are shown, other numbers and types of user computing systems could be used. In one example, the client devices 14(1)-14(n) comprise mobile devices with Internet access that permit a website form page or other retrieved data that is a part of a requested network resource to be displayed, although each of the client devices 14(1)-14(n) can connect to server devices 16(1)-16(n) via other types of network connections directly or indirectly, depending upon specific scenarios, as can be contemplated by one of ordinary skill in the art, after reading this disclosure. By way of example only, one or more of the client devices 14(1)-14(n) can comprise smart phones, personal digital assistants, computers, or other computing devices.

[00021] Each of client devices 14(1)-14(n) in this example is a computing device that includes a central processing unit (CPU) or processor 20, a memory 22, user input device 24, a display 26, and an interface system 28, and which are coupled together by a bus 30 or other link, although one or more of client devices 14(1)-14(n) can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 20 in each of client

devices 14(1)-14(n) executes a program of stored instructions for aiding one or more aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions.

[00022] The memory 22 in each of the client devices 14(1)-14(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein as well as other data, such as updated cookies associated with a network resource and received as part of one or more redirect chains forwarded by web content optimization apparatus 12, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor 20 can be used for the memory 22 in each of the client devices 14(1)-14(n).

[00023] The user input device 24 in each of the client devices 14(1)-14(n) is used to input selections, such as requests for a network resource, *e.g.*, a particular website form page or to enter data in fields of a form page, although the user input device could be used to input other types of data and interact with other elements of exemplary environment 10. The user input device 24 can include keypads, touch screens, and/or vocal input processing systems, although other types and numbers of user input devices can be used.

[00024] The display 26 in each of the client devices 14(1)-14(n) is used to show data and information to the user, such as requested website page by way of example only. The display in each of the client devices 14(1)-14(n) is a mobile phone screen display, although other types and numbers of displays could be used depending on the particular type of client device, as can be contemplated by one of ordinary skill in the art, after reading this disclosure.

[00025] The interface system 28 in each of the client devices 14(1)-14(n) is used to operatively couple and communicate between the client devices 14(1)-

14(n) and the web content optimization apparatus 12 and server devices 16(1)-16(n) over the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00026] The server devices 16(1)-16(n) provide one or more pages from one or more web sites for use by one or more of the client devices 14(1)-14(n) via the web content optimization apparatus 12, although the server devices 16(1)-16(n) can provide other numbers and types of applications and/or content and can have provide other numbers and types of functions. Although server devices 16(1)-16(n) are shown for ease of illustration and discussion, other numbers and types of server systems, for example, web servers, and devices can be used. In one example, server devices 16(1)-16(n) can be web servers having dedicated hardware with software executing on the dedicated hardware to facilitate the web content optimization apparatus 12 and client devices 14(1)-14(n) in their functioning. In another example, server devices 16(1)-16(n) can be content servers that are configured to deliver network resources stored thereupon using the HTTP protocol, or other network protocols for example. Content stored on server devices 16(1)-16(n) that can be part of the network resources requested by client devices 14(1)-14(n) can be web pages, electronic files and documents, configuration data, metadata, or other network data and files, by way of example only and not by way of limitation.

[00027] Each of the server devices 16(1)-16(n) include a central processing unit (CPU) or processor, a memory, and an interface system which are coupled together by a bus or other link, although each of the server devices 16(1)-16(n) could have other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor in each of the server devices 16(1)-16(n) executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions. When one of the server devices 16(1)-16(n) does not store the requested content, the server device may respond by sending a redirect message to the web content optimization apparatus 12.

[00028] The memory in each of the server devices 16(1)-16(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated by way of the embodiments, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor, can be used for the memory in each of the server devices 16(1)-16(n).

[00029] The interface system in each of the server devices 16(1)-16(n) is used to operatively couple and communicate between the server devices 16(1)-16(n) and the web content optimization apparatus 12 and the client devices 14(1)-14(n) via communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00030] Although embodiments of the web content optimization apparatus 12, the client devices 14(1)-14(n), and the server devices 16(1)-16(n), are described and illustrated herein, each of the client devices 14(1)-14(n), the web content optimization apparatus 12, and the server devices 16(1)-16(n), can be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s).

[00031] Furthermore, each of the systems of the embodiments may be conveniently implemented using one or more general purpose computer systems having non-transitory computer readable medium, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[00032] In addition, two or more computing systems or devices can be substituted for any one of the systems in any embodiment of the embodiments. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the embodiments. The embodiments may also be implemented on computer system or systems that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (*e.g.*, voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[00033] The embodiments may also be embodied as non-transitory computer readable medium having instructions stored thereon for one or more aspects of the present invention as described and illustrated by way of the embodiments herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the embodiments, as described and illustrated herein.

[00034] Referring to FIG. 2, an exemplary method 200 for processing an exemplary sequence of HTTP requests and responses between one of the client devices 14(1)-14(n) and the server devices 16(1)-16(n) hosting example.com and sample.com without a the web content optimization apparatus is illustrated. It is to be noted that the sequence of steps for the method 200 is only exemplary and one of ordinary skill in the art, after reading this disclosure, can contemplate alternative sequences of steps that achieve substantially the same result.

[00035] More specifically, at step 202 one of the client devices 14(1)-14(n) (*e.g.*, a mobile device) sends an HTTP request: GET/A.html HTTP/1.1 for a network resource at a Uniform Resource Locator (URL) http://www.example.com/A.html, although other types of requests for other types of network resources may be sent. Although in this example one of the client

devices 14(1)-14(n) via a web browser requests a page A.html at the URL, "www.example.com" as shown, by way of example only, client devices 14(1)-14(n) may send a request for a network shared data file using a file transfer protocol instead of a URL using the HTTP protocol.

[00036] In step 204, the requesting one of the client devices 14(1)-14(n) gets a response from the one of the server devices 16(1)-16(n) that was previously hosting the network resource, in this example the requested page A.html. The response includes a temporary redirect message (shown as an exemplary status code 302 with a message "Moved Temporarily") to a different URL http://sample.com/B.html along with a "Set-Cookie" including a cookie named "SESSION" set to value "1234".

[00037] Generally, the cookie is a string formed by the pair "name=value" (*e.g.*, "SESSION=1234", followed by optional attributes. Although one illustrative example is described herein, this technology can be used with specifications for all cookies.

[00038] In step 206, the requesting one of the client devices 14(1)-14(n) processes the response and sends a new HTTP request to the one of the server devices 16(1)-16(n) hosting the URL http://www.sample.com/B.html.

[00039] In step 208, the requesting one of the client devices 14(1)-14(n) gets a response from the one of the server devices 16(1)-16(n) that was previously hosting the network resource, in this example the requested page B.html. The response includes another temporary redirect message, "Moved Temporarily" to a different URL http://example.com/C.html along with a "Set-Cookie" including a cookie named "LANG" set to value "EN".

[00040] In step 210, the requesting one of the client devices 14(1)-14(n) processes the response and sends another new HTTP request to the one of the server devices 16(1)-16(n) hosting the URL http://www.example.com/C.html.

[00041] In step 212, the requesting one of the client devices 14(1)-14(n) gets a response from the one of the server devices 16(1)-16(n) hosting the network

resource with a status identifier message shown as status code 200 set to "OK" and the network resource can now be retrieved. Although exemplary steps 202-212 are shown, a higher or a lower number of redirections may be required by a particular request.

[00042] Referring to FIG. 3, a flow chart of a method for processing an exemplary sequence of HTTP requests and responses between one of the client devices 14(1)-14(n) and the web content optimization apparatus 12 is illustrated. In step 302, one of the client devices 14(1)-14(n) makes a request for http://m.acme.com/m/example.com/A.html which is received by the web content optimization apparatus 12.

[00043] In step 304, the web content optimization apparatus 12 generates a request for A.html at example.com which is transmitted to the one of the server devices 16(1)-16(n) thought to host the network resource.

[00044] In step 306, the web content optimization apparatus 12 receives a redirect response from the one of the server devices 16(1)-16(n) to http://sample.com/B.html and setting the cookie SESSION=1234 which is provided to the requesting one of the client devices 14(1)-14(n).

[00045] In step 308, this redirect response is optimized by the web content optimization apparatus 12 into a redirect to http://m.acme.com/m/sample.com/B.html with optimized version cookie SESSION+example.com+%2F=1234 in this example and transmitted to the requesting one of the client devices 14(1)-14(n).

[00046] In step 310, the requesting one of the client devices 14(1)-14(n) makes the next request for http://m.acme.com/m/sample.com/B.html based on the received optimized redirect response. This request is received by the web content optimization apparatus 12.

[00047] In step 312 the web content optimization apparatus 12 generates a request for document B.html on sample.com which is transmitted to the one of the server devices 16(1)-16(n) thought to currently host the network resource.

[00048] In step 314, the web content optimization apparatus 12 receives a redirect response from the one of the server devices 16(1)-16(n) to http://example.com/C.html setting the cookie LANG=EN.

[00049] In step 316, this redirect response is optimized by the web content optimization apparatus 12 into a redirect to http://m.acme.com/m/example.com/C.html with optimized version cookie LANG+sample.com+%2F=EN which is transmitted to the requesting one of the client devices 14(1)-14(n).

[00050] In step 318, the one of the client devices 14(1)-14(n) makes the next request for http://m.acme.com/m/example.com/C.html which is received by the web content optimization apparatus 12.

[00051] In step 320 the web content optimization apparatus 12 generates a request for document C.html on example.com passing the matching cookie SESSION=1234 to the one of the server devices 16(1)-16(n) thought to currently host the network resource.

[00052] In step 322, gets a response from the one of the server devices 16(1)-16(n) with a status identifier message shown as status code 200 set to "OK" indicating the network resource has been found. In step 324, the web content optimization apparatus 12 provides this response to the requesting one of the client devices 14(1)-14(n) to retrieve the located network resource.

[00053] Referring to FIG. 4, an exemplary method 400 for processing an exemplary sequence of HTTP requests and responses between one of the client devices 14(1)-14(n) and the web content optimization apparatus 12 when auto redirect is enabled. In step 402, one of the client devices 14(1)-14(n), which in this example is a mobile device, makes a new request for the URL: m.acme.com/m/example.com/A.html which is received by the web content optimization apparatus 12.

[00054] In step 404, the web content optimization apparatus 12 processes the request utilizing processors 13(1) and 13(2) which act as both the requesting client device and the proxy server device as explained in greater detail below.

[00055] More specifically, in step 404a one of the processors 13(1) in the web content optimization apparatus 12 acting as the client device sends a request for /m/example.com/A.html to another one of the processors 13(2) in the web content optimization apparatus 12. The auto redirect between the processors 13(1) and 13(2) in the web content optimization apparatus 12 is turned off.

[00056] In step 404b, the other one of the processors 13(2) in the web content optimization apparatus 12 sends the request to one of the server devices 16(1)-16(n) currently identified as hosting the requested resource. The processor 13(2) in the web content optimization apparatus 12 gets a redirect response from one of the server devices 16(1)-16(n) thought to store the network resource. The processor 13(2) in the web content optimization apparatus 12 optimizes the redirect response to http://m.acme/m/sample.com/B.html, and cookie to SESSION+example.com+%2F=1234, which is provided to the processor 13(1) in the web content optimization apparatus 12 acting as the requesting client device.

[00057] In step 404c, based on the redirect response the processor 13(1) in the web content optimization apparatus 12 acting as the requesting client device sends an updated HTTP request to the processor 13(2) in the web content optimization apparatus 12 acting as the proxy server device. The processor 13(2) in the web content optimization apparatus 12 sends the updated request to the one of the server devices 16(1)-16(n) identified as hosting the network resource.

[00058] In step 404d, the processor 13(2) in the web content optimization apparatus 12 gets a redirect response from the one of the server devices 16(1)-16(n) identified as hosting the network resource based on the updated request. The processor 13(2) in the web content optimization apparatus 12 optimizes the redirect response to http://m.acme/m/sample.com/C.html, and the cookie LANG+sample.com+%2F=EN, which is provided to the processor 13(1) in the web content optimization apparatus 12 acting as the requesting client device.

[00059] In step 404e, based on the last redirect response the processor 13(1) in the web content optimization apparatus 12 acting as the requesting client device sends another updated HTTP request to the processor 13(2) in the web content optimization apparatus 12 acting as the proxy server device. The processor 13(2) in the web content optimization apparatus 12 sends the updated request to the one of the server devices 16(1)-16(n) identified as currently hosting the network resource.

[00060] In step 404f, the processor 13(2) in the web content optimization apparatus 12 gets a response from the one of the server devices 16(1)-16(n) with a status identifier message shown as status code 200 set to "OK" indicating the network resource has been found.

[00061] In step 406, when the processor 13(1) in the web content optimization apparatus 12 gets the http://m.acme.com/m/example.com/C.html response that is a 200 (i.e. a real resource), then it creates the http response for the requesting one of the client devices 14(1)-14(n) that is a redirect to http://m.acme.com/m/example.com/C.html with the two collected cookies from steps 404b and 404d.

[00062] Referring to FIG. 5, an exemplary method for reducing latency in network connections between requesting client devices 14(1)-14(n) and content hosting server devices 16(1)-16(n) is described using flowchart 500 with reference back to FIGS. 1 and 3-4. More specifically, details of operation of web content optimization apparatus 12 with respect to optimization of redirect chain 300 and 400 using core module 21 and redirect module 23 are described. Redirect module 23 is part of a web content optimization system (referred to as "WCOS," not shown) implemented in web content optimization apparatus 12 that is responsible for handling multiple content server redirect response messages from server devices 16(1)-16(n) and sending them to client devices 14(1)-14(n) that originally made the HTTP request (*e.g.*, as shown in steps 202 and 302). In one example, only the last URL of the redirect chain 300 is handled by the redirect module 23. Core module 21 and redirect module 23 reside as part of WCOS on hardware of memory 15. Alternatively, core module 21 and 23 may be implemented at least

partially in software residing on a hardware portion of memory 15. The embodiment described in FIG. 5 adds an additional layer of optimization to handling redirect chains 300 and 400 by operating upon prior already optimized requests.

[00063] The method of flowchart 500 begins in step 502 where redirect module 23 inside web content optimization apparatus 12 receives a request from core module 21, which received request was a part of a request originally received by the web content optimization apparatus 12 from the requesting one of the client devices 14(1)-14(n) (*e.g.*, from a web browser on one of the client devices 14(1)-14(n)).

[00064] In step 504, redirect module 23 determines whether an automatic-redirect identifier or flag is enabled and was received as part of the original request for the network resource by the requesting one of the client devices 14(1)-14(n). The automatic-redirect, also referred to as "auto-redirect" herein, is a programming instruction that is used to decide which HTTP requests from client devices 16(1)-16(n) should be optimized for handling by the web content optimization apparatus 12, in accordance with the redirect chain 400 described by FIG. 4, and which requests should be handled directly by the requesting client devices 14(1)-14(n) directly, as shown by FIG. 2 with minimum processing by the web content optimization apparatus 12, although other options could be provided such as the exemplary redirect illustrated and described with reference to FIG. 3. The auto-redirect flag/identifier adds an additional level of optimization because automatic redirection by web content optimization apparatus 12 is used only when needed and when chosen by the programmer or administrator of the web content optimization apparatus 12. If the auto-redirect flag/identifier is not enabled or set to a value "false" set programmatically by a WCOS programmer, the No branch is taken in step 504, where the request is returned to core module 21 by redirect module 23, as shown in step 518. In step 518, the web content optimization apparatus 12 can process the request from the requesting one of the client devices 14(1)-14(n), send a request to one of the server devices 16(1)-16(n), and then optimize the response which is sent back to the requesting one of the client devices 14(1)-14(n).

[00065] However, if the auto-redirect flag/parameter/identifier is set to a value "true," the Yes branch is taken and the flow proceeds to step 506. In step 506, the same HTTP request is sent back to the core module 21 in WCOS with its "auto-redirect" flag set to a value "false." By way of example only, m.acme.com is the host name associated with the web content optimization system residing in memory 15 of the web content optimization apparatus 12. When the auto-redirect parameter is disabled and m.acme.com receives an exemplary request http://m.acme.com/example.com/A.html, it sends a request to http://example.com/A.html getting a redirect to http://sample.com/B.html with SESSION=1234 cookie (as shown, for example, in FIG. 2). Accordingly, m.acme.com will send response to the client device with a redirect to http://m.acme.com/sample.com/B.html with cookie modified to SESSION+example.com+%2F=1234. Modification of the cookie is explained in more detail, by way of example only, in U.S. Patent Application Serial No. 12/660,637, filed March 2, 2010, entitled "Method for Optimizing a Web Content Proxy Server and Devices Thereof," hereby incorporated by reference in its entirety. When core module 21 and redirect module 23 receive a request for http://m.acme.com/sample.com/B.html from the client device (*e.g.*, client device 14(1)), a request to http://sample.com/B.html is sent and then a redirect to http://example.com/C.html with LANG=EN cookie is received that will be transformed in a redirect response to http://m.acme.com/example.com/C.html with LANG+sample.com+%2F=EN cookie, to be sent to the client web browser of the client device.

[00066] In another exemplary scenario, automatic-redirect is enabled for first request for http://m.acme.com/example.com/A.html. Since automatic-redirect is enabled, the redirect module 23 will decide to follow the redirect chain 300 making the HTTP requests internally to itself within the web content optimization apparatus 12, including all the intermediate redirection requests. Accordingly, redirect module 23 will make HTTP request to http//m.acme.com/example.com/A.html with auto redirect disabled. This way, correct optimized cookies will be collected by redirect module 23 and WCOS redirects will be followed. When all the redirect chain 400 is collected as

illustrated and described with reference to FIG. 4, a redirect response to http://m.acme.com/example.com/C.html will be sent to the client browser having new cookies set to SESSION+example.com+%2F=1234 and LANG+sample.com+%2F=EN that are valid cookies for m.acme.com domain.

[00067] In step 508, the HTTP response is received by core module 21. Sending HTTP requests back to core module 21 of WCOS instead of to hosting one of the server devices 16(1)-16(n) enables the WCOS to optimize the redirect chain 400 of already optimized HTTP responses, although again other optimized redirect chains can be used as illustrated and described with reference to FIG. 3 by way of example. In some examples, core module 21 of WCOS can create longer redirect chains than original content site. By way of example only, the web content optimization system adds new redirect chains to the original server request-response flow. For example, when the web content optimization system has to simulate a complex JavaScript behavior that is available on the original web site page for JavaScript enabled browsers, on its server side JavaScript emulator. Using the auto redirect module 23 over already optimized WCOS pages can advantageously improve the WCOS performance in the web content optimization apparatus 12.

[00068] In step 510, if the response from the redirect module 23 in WCOS optimization system is a redirect response message, then the Yes branch is taken to step 512.

[00069] In step 512, a new HTTP request is created for the redirect URL and the auto-redirect flag is set to a value "false." Further, by way of example only, cookies in the header field are the result of merging the cookies set by the redirect responses with the cookies passed to the first HTTP request.

[00070] In step 514, the core module 21 will then create the appropriate HTTP response to be sent to the redirect module 23. If the HTTP response from the redirect module 23 is a redirect response, the previous steps 510-514 will be repeated one or more times.

[00071] However, in step 510, if the response is a real resource, then the No branch is taken to step 516 where the last redirect response coupled with collected response cookies in the header fields are sent back to the core module 21. By way of example only, the method of flowchart 500 is used when the requesting one of the client devices 14(1)-14(n) is a mobile client device, for example, a cellular telephone.

[00072] Accordingly, as illustrated and described herein this technology provides a number of advantages including providing a method, computer readable medium and an apparatus that reduces latency in network connections, for example, by managing handling of redirect messages from server devices 16(1)-16(n) substantially at web content optimization apparatus 12 interposed between the client devices 14(1)-14(n) and server devices 16(1)-16(n) on which various network resources are stored, which network resources are requested by client devices 14(1)-14(n). With this technology, the web content optimization apparatus 12 optimizes network resource provisioning by returning only the last redirect message along with all cookies collected from intermediate redirect messages from a chain of redirect messages from server devices 16(1)-16(n) to client devices 14(1)-14(n) for a particular network resource, thereby reducing latency in network connections between client devices and server devices.

[00073] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for reducing latency in network connections, the method comprising:

receiving at an optimization apparatus a request from one of one or more client devices for network content hosted at one of one or more server devices;

determining at the optimization apparatus whether an automatic redirect parameter is enabled for the request;

sending the request from one proxy processing device acting as the requesting one of the client devices within the optimization apparatus to another proxy processing device within the optimization apparatus when the automatic redirect parameter is determined to be enabled;

obtaining one or more redirect messages with one or more optimized cookies associated with the requested network resource from the one or more server devices at the another proxy processing device within the optimization apparatus;

sending the last of the one or more redirects messages with the one or more optimized cookies when the requested network resource is located from the another proxy processing device within the optimization apparatus to the one proxy processing device within the optimization apparatus; and

forwarding the last of the one or more redirect messages with the one or more optimized cookies associated with the requested network resource from the optimization apparatus to the requesting one of the one or more of client devices.

2. The method as set forth in claim 1 wherein the determining at the optimization apparatus whether the automatic redirect parameter is enabled for the request is based on information contained in the request.

3. The method as set forth in claim 1 further comprising disabling the automatic redirect parameter when the request is sent from the one

proxy processing device acting as the requesting one of the client devices within the optimization apparatus to the another proxy processing device within the optimization apparatus.

4. The method as set forth in claim 1 further comprising:

processing at the web content optimization apparatus a revised request for the requested network resource from the requesting one of the plurality of client devices based on the last of the one or more redirect messages with the one or more optimized cookies;

obtaining network content at the optimization apparatus from one of the one or more server devices based on the revised request;

optimizing at the optimization apparatus the obtained network content at the web content optimization apparatus for the requesting one of the client devices; and

forwarding from the optimization apparatus the optimized obtained network content to the requesting one of the one or more of the client devices.

5. The method as set forth in claim 1, wherein the receiving further comprises receiving at the optimization apparatus the request from the one of the one or more of the client devices comprising a mobile device configured to display the optimized network content on a browser of the mobile device.

6. The method as set forth in claim 2, wherein the information contained in the request comprises a Uniform Resource Identifier (URI).

7. The method as set forth in claim 1, wherein the network content comprises one or more web pages.

8. A non-transitory computer readable medium having stored thereon instructions for reducing latency in network connections comprises

machine executable code which when executed by at least one processor, causes the processor to perform steps comprising:

receiving a request from one of one or more client devices for network content hosted at one of one or more server devices;

determining whether an automatic redirect parameter is enabled for the request;

sending the request from one proxy processing device acting as the requesting one of the client devices within an optimization apparatus to another proxy processing device within the optimization apparatus when the automatic redirect parameter is determined to be enabled;

obtaining one or more redirect messages with one or more optimized cookies associated with the requested network resource from the one or more server devices at the another proxy processing device within the optimization apparatus;

sending the last of the one or more redirects messages with the one or more optimized cookies when the requested network resource is located from the another proxy processing device within the optimization apparatus to the one proxy processing device within the optimization apparatus; and

forwarding the last of the one or more redirect messages with the one or more optimized cookies associated with the requested network resource to the requesting one of the one or more of client devices.

9. The medium as set forth in claim 8 wherein the determining at the optimization apparatus whether the automatic redirect parameter is enabled for the request is based on information contained in the request.

10. The medium as set forth in claim 8 further comprising disabling the automatic redirect parameter when the request is sent from the one proxy processing device acting as the requesting one of the client devices within the optimization apparatus to the another proxy processing device within the optimization apparatus.

11. The medium as set forth in claim 8 further comprising:

processing a revised request for the requested network resource from the requesting one of the plurality of client devices based on the last of the one or more redirect messages with the one or more optimized cookies;

obtaining network content from one of the one or more server devices based on the revised request;

optimizing the obtained network content at the optimization apparatus for the requesting one of the client devices; and

forwarding the optimized obtained network content to the requesting one of the one or more of the client devices.

12. The medium as set forth in claim 8, wherein the receiving further comprises receiving the request from the one of the one or more of the client devices comprising a mobile device configured to display the optimized network content on a browser of the mobile device.

13. The medium as set forth in claim 9, wherein the information contained in the request comprises a Uniform Resource Identifier (URI).

14. The medium as set forth in claim 8, wherein the network content comprises one or more web pages.

15. An apparatus configured to reduce latency in network connections comprising one or more processors and a memory coupled to the one or more processors configured to execute programmed instructions stored in the memory comprising:

receiving a request from one of one or more client devices for network content hosted at one of one or more server devices;

determining whether an automatic redirect parameter is enabled for the request;

sending the request from one proxy processing device acting as the requesting one of the client devices within an optimization apparatus

to another proxy processing device within the optimization apparatus when the automatic redirect parameter is determined to be enabled;

obtaining one or more redirect messages with one or more optimized cookies associated with the requested network resource from the one or more server devices at the another proxy processing device within the optimization apparatus;

sending the last of the one or more redirects messages with the one or more optimized cookies when the requested network resource is located from the another proxy processing device within the optimization apparatus to the one proxy processing device within the optimization apparatus; and

forwarding the last of the one or more redirect messages with the one or more optimized cookies associated with the requested network resource to the requesting one of the one or more of client devices.

16. The apparatus as set forth in claim 15 wherein the one or more processors is further configured to execute programmed instructions stored in the memory for the determining further comprising determining at the optimization apparatus whether the automatic redirect parameter is enabled for the request based on information contained in the request.

17. The apparatus as set forth in claim 15 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising disabling the automatic redirect parameter when the request is sent from the one proxy processing device acting as the requesting one of the client devices within the optimization apparatus to the another proxy processing device within the optimization apparatus.

18. The apparatus as set forth in claim 15 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising:

processing a revised request for the requested network resource from the requesting one of the plurality of client devices based on the last of the one or more redirect messages with the one or more optimized cookies;

obtaining network content from one of the one or more server devices based on the revised request;

optimizing the obtained network content at the optimization apparatus for the requesting one of the client devices; and

forwarding the optimized obtained network content to the requesting one of the one or more of the client devices.

19. The apparatus as set forth in claim 15, wherein the one or more processors is further configured to execute programmed instructions stored in the memory for the receiving further comprising receiving the request from the one of the one or more of the client devices comprising a mobile device configured to display the optimized network content on a browser of the mobile device.

20. The apparatus as set forth in claim 16, wherein the information contained in the request comprises a Uniform Resource Identifier (URI).

21. The apparatus as set forth in claim 15, wherein the network content comprises one or more web pages.

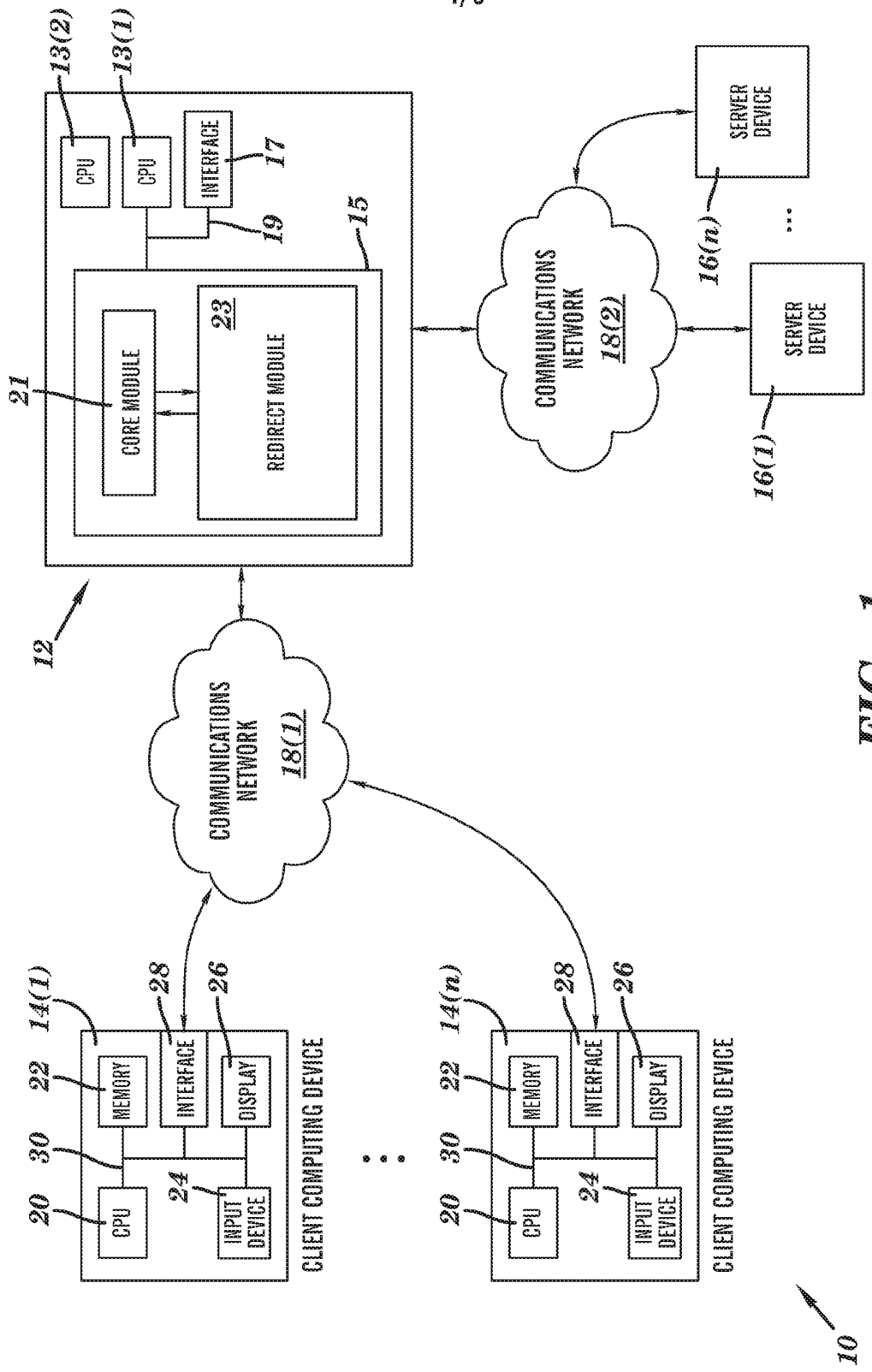

13(2)
13(1)
CPU
CPU
INTERFACE
17
19
15
21
CORE MODULE
23
REDIRECT MODULE
12
SERVER DEVICE
16(n)
...
SERVER DEVICE
16(1)
COMMUNICATIONS NETWORK
18(2)
COMMUNICATIONS NETWORK
18(1)
14(1)
28
26
22
MEMORY
INTERFACE
DISPLAY
30
20
CPU
24
INPUT DEVICE
CLIENT COMPUTING DEVICE
14(n)
28
26
22
MEMORY
INTERFACE
DISPLAY
30
20
CPU
24
INPUT DEVICE
CLIENT COMPUTING DEVICE
10

FIG. 1

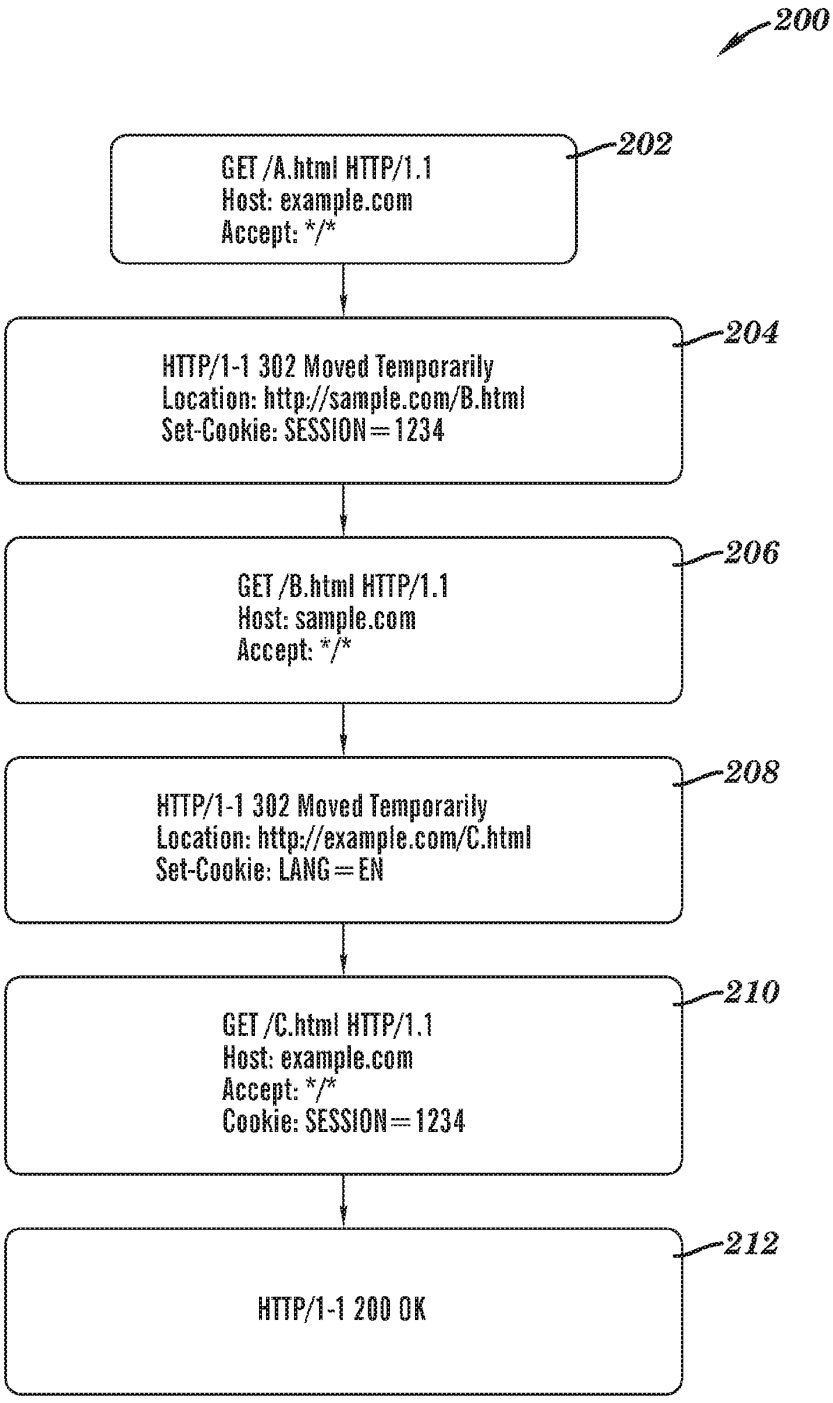

200
202
GET /A.html HTTP/1.1
Host: example.com
Accept: */*
204
HTTP/1-1 302 Moved Temporarily
Location: http://sample.com/B.html
Set-Cookie: SESSION=1234
206
GET /B.html HTTP/1.1
Host: sample.com
Accept: */*
208
HTTP/1-1 302 Moved Temporarily
Location: http://example.com/C.html
Set-Cookie: LANG=EN
210
GET /C.html HTTP/1.1
Host: example.com
Accept: */*
Cookie: SESSION=1234
212
HTTP/1-1 200 OK

FIG. 2

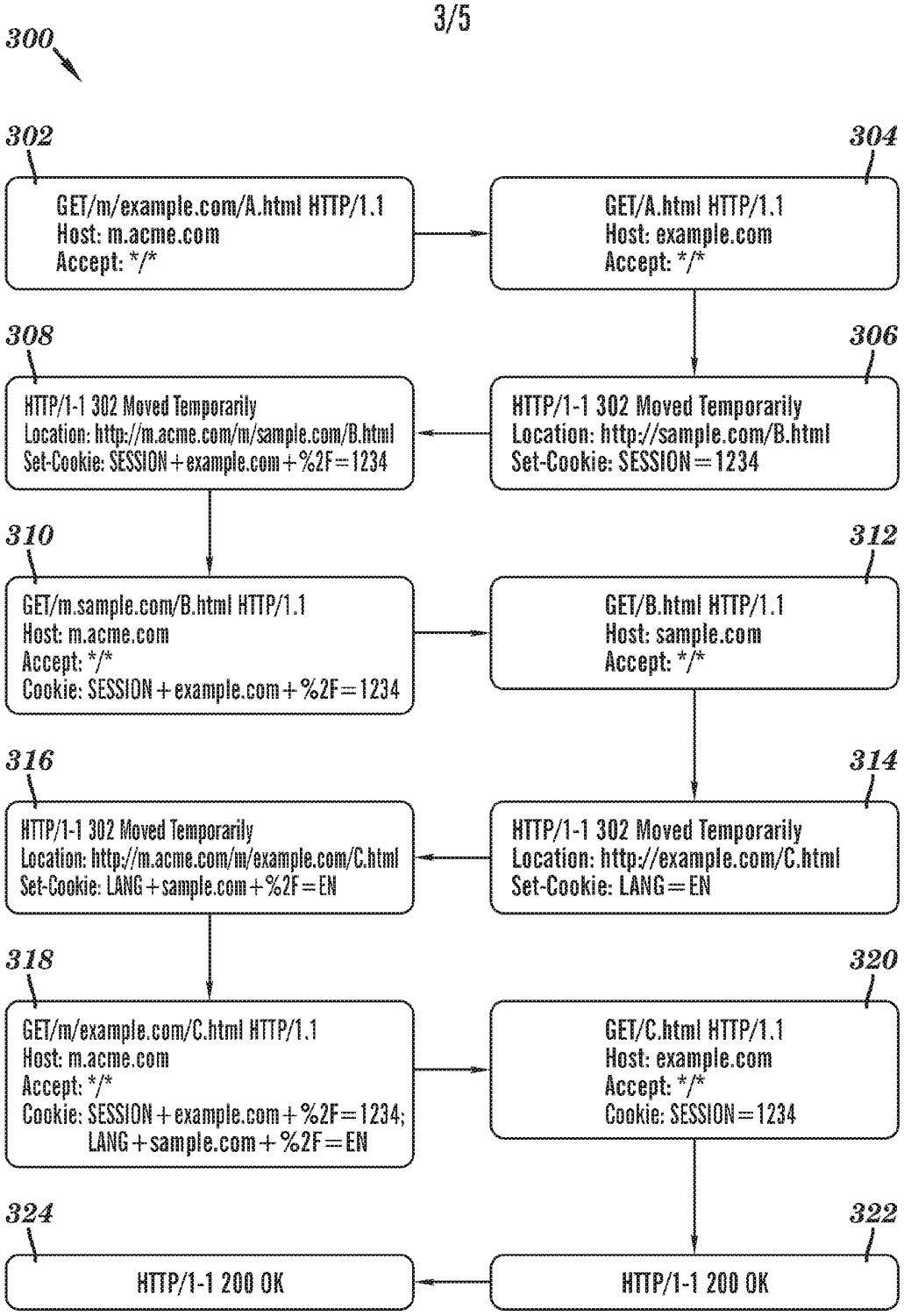

300
302
GET/m/example.com/A.html HTTP/1.1
Host: m.acme.com
Accept: */*
304
GET/A.html HTTP/1.1
Host: example.com
Accept: */*
308
HTTP/1-1 302 Moved Temporarily
Location: http://m.acme.com/m/sample.com/B.html
Set-Cookie: SESSION+example.com+%2F=1234
306
HTTP/1-1 302 Moved Temporarily
Location: http://sample.com/B.html
Set-Cookie: SESSION=1234
310
GET/m.sample.com/B.html HTTP/1.1
Host: m.acme.com
Accept: */*
Cookie: SESSION+example.com+%2F=1234
312
GET/B.html HTTP/1.1
Host: sample.com
Accept: */*
316
HTTP/1-1 302 Moved Temporarily
Location: http://m.acme.com/m/example.com/C.html
Set-Cookie: LANG+sample.com+%2F=EN
314
HTTP/1-1 302 Moved Temporarily
Location: http://example.com/C.html
Set-Cookie: LANG=EN
318
GET/m/example.com/C.html HTTP/1.1
Host: m.acme.com
Accept: */*
Cookie: SESSION+example.com+%2F=1234;
LANG+sample.com+%2F=EN
320
GET/C.html HTTP/1.1
Host: example.com
Accept: */*
Cookie: SESSION=1234
324
HTTP/1-1 200 OK
322
HTTP/1-1 200 OK

FIG. 3

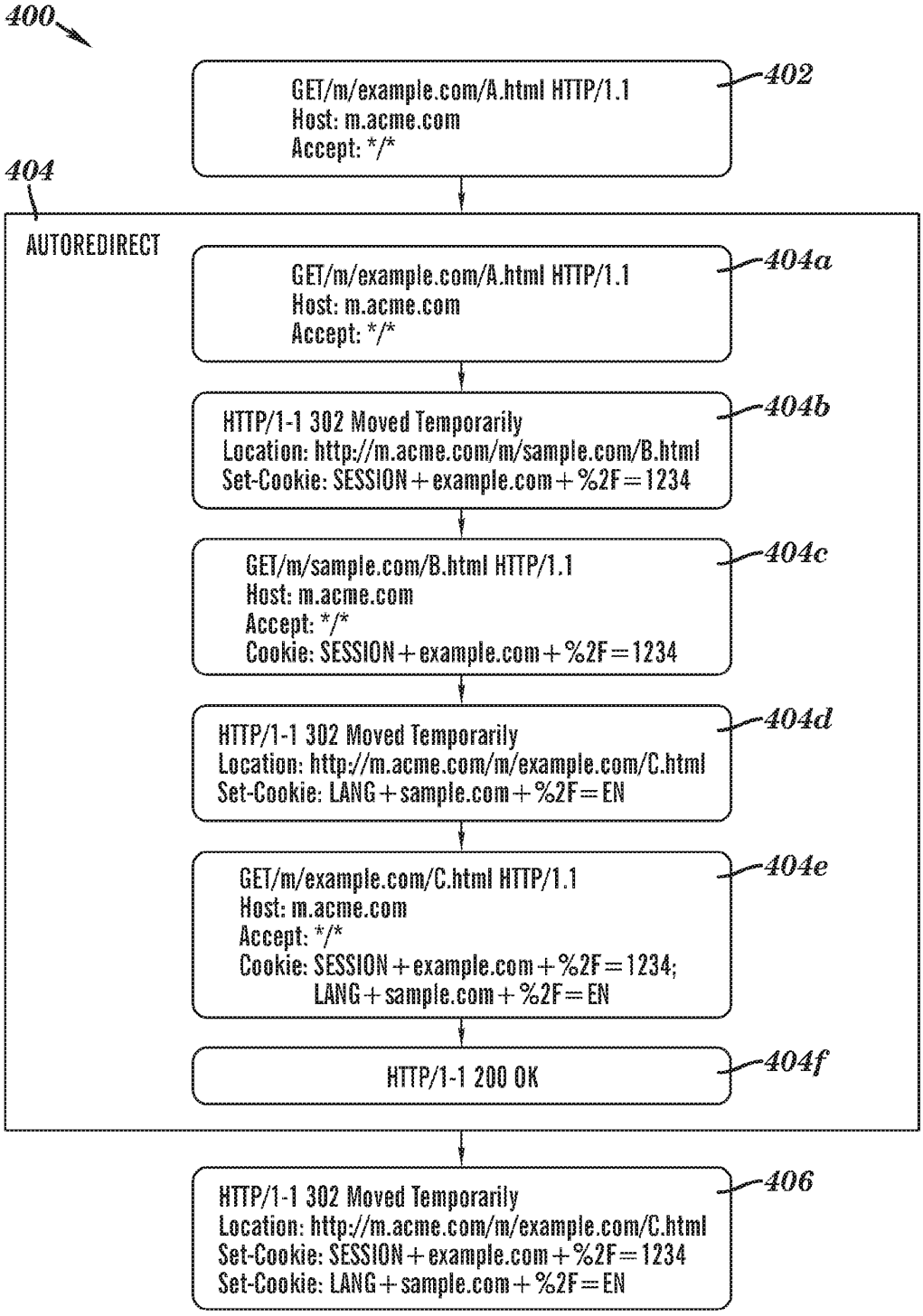

400
GET/m/example.com/A.html HTTP/1.1
Host: m.acme.com
Accept: */*
402
404
AUTOREDIRECT
GET/m/example.com/A.html HTTP/1.1
Host: m.acme.com
Accept: */*
404a
HTTP/1-1 302 Moved Temporarily
Location: http://m.acme.com/m/sample.com/B.html
Set-Cookie: SESSION+example.com+%2F=1234
404b
GET/m/sample.com/B.html HTTP/1.1
Host: m.acme.com
Accept: */*
Cookie: SESSION+example.com+%2F=1234
404c
HTTP/1-1 302 Moved Temporarily
Location: http://m.acme.com/m/example.com/C.html
Set-Cookie: LANG+sample.com+%2F=EN
404d
GET/m/example.com/C.html HTTP/1.1
Host: m.acme.com
Accept: */*
Cookie: SESSION+example.com+%2F=1234;
LANG+sample.com+%2F=EN
404e
HTTP/1-1 200 OK
404f
HTTP/1-1 302 Moved Temporarily
Location: http://m.acme.com/m/example.com/C.html
Set-Cookie: SESSION+example.com+%2F=1234
Set-Cookie: LANG+sample.com+%2F=EN
406

FIG. 4

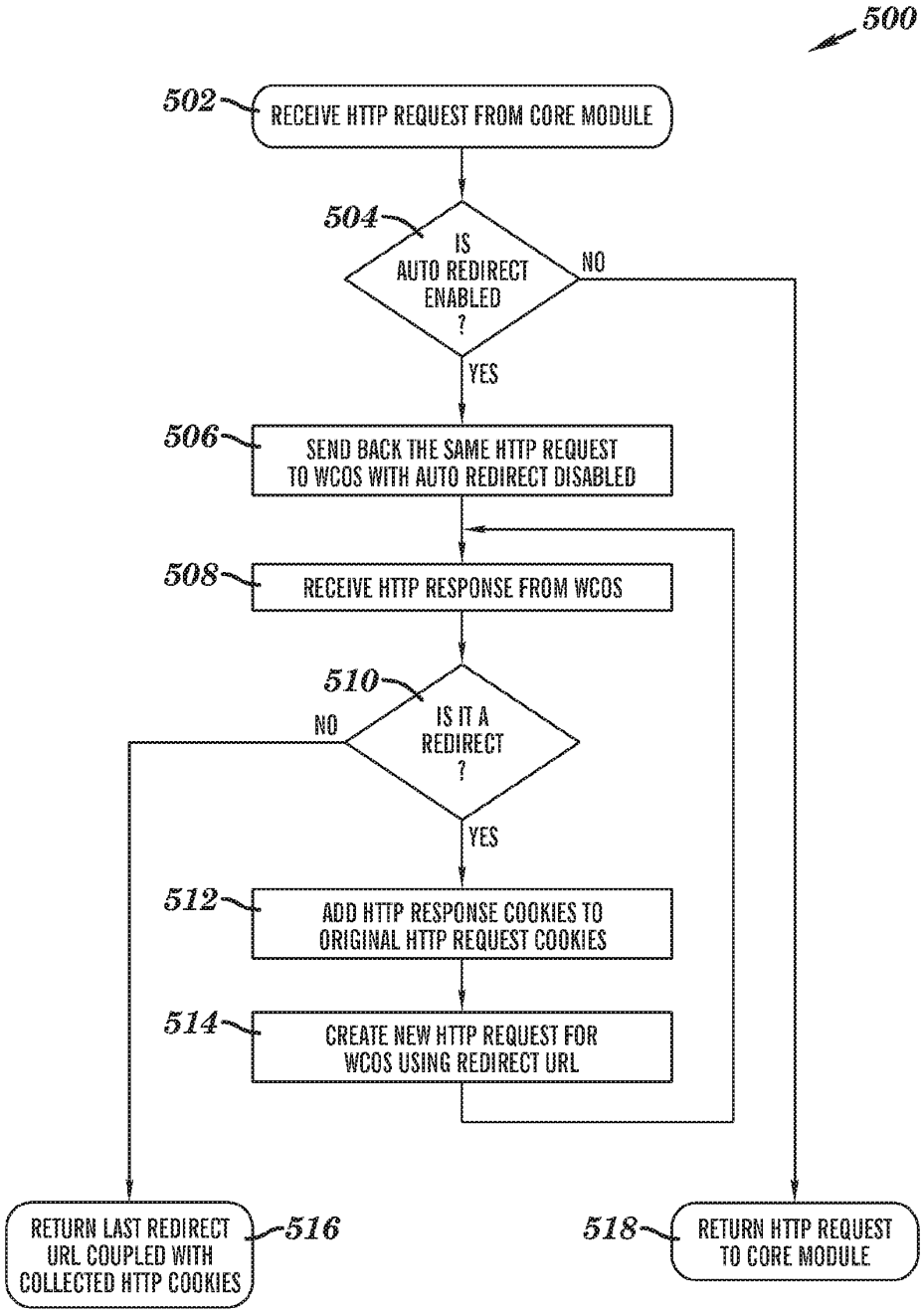

500
502
RECEIVE HTTP REQUEST FROM CORE MODULE
504
IS AUTO REDIRECT ENABLED ?
NO
YES
506
SEND BACK THE SAME HTTP REQUEST TO WCOS WITH AUTO REDIRECT DISABLED
508
RECEIVE HTTP RESPONSE FROM WCOS
510
IS IT A REDIRECT ?
NO
YES
512
ADD HTTP RESPONSE COOKIES TO ORIGINAL HTTP REQUEST COOKIES
514
CREATE NEW HTTP REQUEST FOR WCOS USING REDIRECT URL
RETURN LAST REDIRECT URL COUPLED WITH COLLECTED HTTP COOKIES
516
518
RETURN HTTP REQUEST TO CORE MODULE

FIG. 5

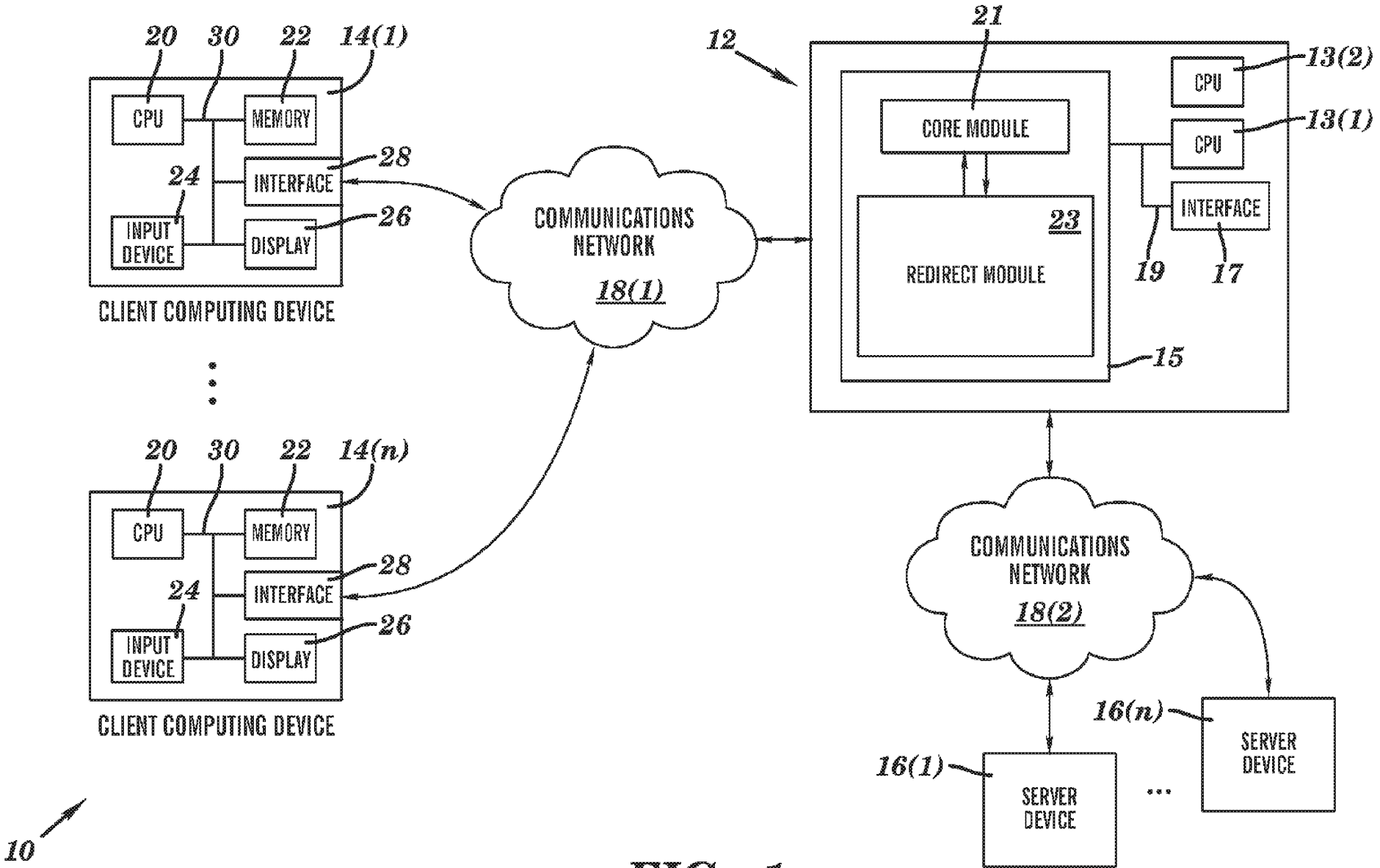


FIG. 1

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2821769 A1 2012/06/28

(21) 2 821 769

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(86) Date de dépôt PCT/PCT Filing Date: 2011/12/21
(87) Date publication PCT/PCT Publication Date: 2012/06/28
(85) Entrée phase nationale/National Entry: 2013/06/13
(86) N° demande PCT/PCT Application No.: US 2011/066611
(87) N° publication PCT/PCT Publication No.: 2012/088326
(30) Priorité/Priority: 2010/12/22 (US12/975,990)

(51) Cl.Int./Int.Cl. *G06Q 50/32* (2012.01)
(71) Demandeur/Applicant:
USABLENET INC., US

(72) Inventeurs/Inventors:
TAYLOR, JASON, GB;
SPERETTA, MIRCO, US

(74) Agent: GOWLING LAFLEUR HENDERSON LLP

(54) Titre : PROCEDES DE TRANSMISSION PAR COURRIEL D'ETIQUETTES SOUS LA FORME DE FICHIERS DE DONNEES PORTATIFS ET DISPOSITIFS ASSOCIES

(54) Title: METHODS FOR EMAILING LABELS AS PORTABLE DATA FILES AND DEVICES THEREOF

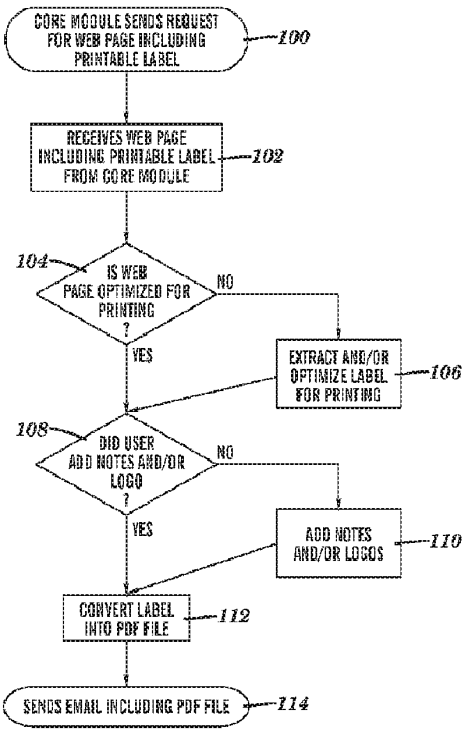


FIG. 3

(57) Abrégé/Abstract:
A method, computer readable medium and device that enables emailing a label as a portable data file to destination information

(21) 2 821 769

(13) A1

(57) Abrégé(suite)/Abstract(continued):
associated with a mobile computing device includes obtaining at a proxy server device a web page with a label from a web server device in response to a request from a mobile computing device. The proxy server device obtains destination information for the label associated with the requesting mobile computing device. The proxy server device converts the label into a portable data file and transmits the portable data file using the destination information for the label associated with the requesting mobile computing device.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

REVISED VERSION

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
28 June 2012 (28.06.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/088326 A9

(51) International Patent Classification:
G06Q 50/32 (2012.01)

(21) International Application Number:
PCT/US2011/066611

(22) International Filing Date:
21 December 2011 (21.12.2011)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/975,990 22 December 2010 (22.12.2010) US

(71) Applicant *(for all designated States except US)*: **US-ABLENET INC.** [US/US]; 28 W. 23rd Street, 6th Floor, New York, NY 10010 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **TAYLOR, Jason** [GB/GB]; 93 North Street, Caine, Wiltshire SN11 0EN (GB). **SPERETTA, Mirco** [IT/US]; 367 Woodridge Avenue, Fairfield, CT 06825 (US).

(74) Agents: **LEINBERG, Gunnar, G.** et al.; LeClairRyan, 70 Linden Oaks, Rochester, NY 14625 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

[Continued on next page]

(54) Title: METHODS FOR EMAILING LABELS AS PORTABLE DATA FILES AND DEVICES THEREOF

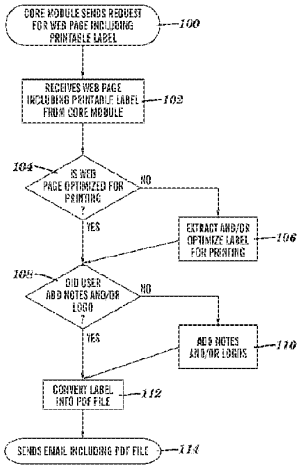


FIG. 3

(57) Abstract: A method, computer readable medium and device that enables emailing a label as a portable data file to destination information associated with a mobile computing device includes obtaining at a proxy server device a web page with a label from a web server device in response to a request from a mobile computing device. The proxy server device obtains destination information for the label associated with the requesting mobile computing device. The proxy server device converts the label into a portable data file and transmits the portable data file using the destination information for the label associated with the requesting mobile computing device.

WO 2012/088326 A9

WO 2012/088326 A9

(88) Date of publication of the revised international search report:

20 September 2012

(15) Information about Correction:
see Notice of 20 September 2012

METHODS FOR EMAILING LABELS AS PORTABLE DATA FILES AND DEVICES THEREOF

FIELD

[0001] This technology relates to methods for emailing labels as portable data files and devices thereof.

BACKGROUND

[0002] Some Web based applications, usually developed for desktop browsers, provide the functionality of printing labels, such as bar codes, proof of insurance, medical records, or bank statements. All these documents have a well-defined printing format, can include personal information and usually are generated from password protected Web content.

[0003] A desktop browser can easily exploit this type of printing functionalities. Unfortunately, mobile browsers do not have the same capability since they usually lack of a direct connection with a printing device.

SUMMARY

[0004] A method for emailing a label as a portable data file to destination information associated with a mobile computing device includes obtaining at a proxy server device a web page with a label from a web server device in response to a request from a mobile computing device. The proxy server device obtains destination information for the label associated with the requesting mobile computing device. The proxy server device converts the label into a portable data file and transmits the portable data file using the destination information for the label associated with the requesting mobile computing device.

[0005] A non-transitory computer readable medium having stored thereon instructions for emailing a label as a portable data file to destination information associated with a mobile computing device comprising machine executable code which when executed by at least one processor, causes the processor to perform steps including obtaining a web page with a label from a web server device in

response to a request from a mobile computing device. Destination information for the label associated with the requesting mobile computing device is obtained. The label is converted into a portable data file and then is transmitted using the destination information for the label associated with the requesting mobile computing device.

[0006] A proxy server device includes one or more processors and a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory including obtaining a web page with a label from a web server device in response to a request from a mobile computing device. Destination information for the label associated with the requesting mobile computing device is obtained. The label is converted into a portable data file and then is transmitted using the destination information for the label associated with the requesting mobile computing device.

[0007] This technology provides a number of advantages including providing a method, computer readable medium and an apparatus that enables a mobile computing device to create a portable data file of one or more labels and send the portable data file, along with custom information, such as descriptions and logos, to a designated email address associated with the mobile computing device. Accordingly, with this technology data files that are not viewable on mobile computing devices can be downloaded and printed.

BRIEF DESCRIPTION OF THE DRAWINGS

[0008] FIG. 1 is a block diagram of an exemplary environment with an exemplary proxy server device;

[0009] FIG. 2 is a screen shot of an exemplary confirmation page with entry fields for data to generate and send a portable data file; and

[00010] FIG. 3 is a flow chart of an exemplary method for generating and emailing labels as portable data files and devices thereof.

DETAILED DESCRIPTION

[00011] An exemplary environment 10 with a proxy server device 12 configured to enable one of the mobile computing devices 14(1)-14(n) to download a printable version of a label that was converted into PDF format is illustrated in FIG. 1. The exemplary environment 10 includes the proxy server device 12, mobile computing devices 14(1)-14(n), web content server devices 16(1)-16(n), and communication networks 18(1)-18(2), although other numbers and types of systems, devices, and/or elements in other configurations and environments with other communication network topologies can be used. This technology provides a number of advantages including providing a method, computer readable medium and an apparatus that enables a mobile computing device to create a portable data file of one or more labels and send the portable data file, along with custom information, such as descriptions and logos, to a designated email address associated with a mobile computing device.

[00012] Referring more specifically to FIG. 1, the proxy server device 12 is a web content optimization device configured to execute the exemplary methods and other technology illustrated and described herein, although the proxy server can comprise other types of computing devices configured to execute the exemplary methods illustrated and described herein. This exemplary proxy server device 12 includes a central processing unit (CPU) or processor 13, a memory 15, and an interface system 17 which are coupled together by a bus 19 or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor 13 in the proxy server device 12 executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[00013] The memory 15 in the proxy server device 12 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different

types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor 13, can be used for the memory 15 in the proxy server device 12.

[00014] The interface system 17 in the proxy server device 12 is used to operatively couple and communicate between the proxy server device 12 and the mobile computing devices 14(1)-14(n), and the web content server devices 16(1)-16(n) via the communication networks 18(1)-18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used. By way of example only, the communication networks 18(1)-18(2) can use TCP/IP over Ethernet and industry-standard protocols, including HTTP, HTTPS, WAP, and SOAP, although other types and numbers of communication networks, such as a direct connection, a local area network, a wide area network, modems and phone lines, e-mail, and wireless and hardwire communication technology, each having their own communications protocols, can be used.

[00015] Each of the mobile computing devices 14(1)-14(n) enables a user to request, get and interact with documents and other files from one or more web sites hosted by the web content server devices 16(1)-16(n) through the proxy server device 12 via one or more communication networks, although one or more of the mobile computing devices 14(1)-14(n) could access content and utilize other types and numbers of applications from other sources and could provide a wide variety of other functions for the user. Although multiple mobile computing devices 14(1)-14(n) are shown, other numbers and types of user computing systems could be used.

[00016] Each of mobile computing devices 14(1)-14(n) in this example is a computing device that includes a central processing unit (CPU) or processor 20, a memory 22, user input device 24, a display 26, and an interface system 28, and which are coupled together by a bus 30 or other link, although one or more of

mobile computing devices 14(1)-14(n) can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 20 in each of mobile computing devices 14(1)-14(n) can execute a program of stored instructions for one or more aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions.

[00017] The memory 22 in each of the mobile computing devices 14(1)-14(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor 20 can be used for the memory 22 in each of the mobile computing devices 14(1)-14(n).

[00018] The user input device 24 in each of the mobile computing devices 14(1)-14(n) is used to input request, selections and other data, although the user input device could provide other functions and interact with other elements. The user input device can include keypads, touch screens, and/or vocal input processing systems although other types and numbers of user input devices can be used.

[00019] The display 26 in each of the mobile computing devices 14(1)-14(n) is used to show data and information to the user, such as a website page optimized for viewing on a mobile computing device by way of example only. The display in each of the mobile computing devices 14(1)-14(n) is a computer screen display, although other types and numbers of displays could be used depending on the particular type of mobile device.

[00020] The interface system 28 in each of the mobile computing devices 14(1)-14(n) is used to operatively couple and communicate between the mobile

computing devices 14(1)-14(n) and the proxy server device 12, and the web content server devices 16(1)-16(n) over the communication networks 18(1)-18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00021] The web content server devices 16(1)-16(n) provide one or more pages from one or more web sites, although the web content server devices 16(1)-16(n) can provide other numbers and types of applications and/or content and can have provide other numbers and types of functions. Although web content server devices 16(1)-16(n) are shown for ease of illustration and discussion, other numbers and types of web server systems and devices can be used.

[00022] Each of the web content server devices 16(1)-16(n) include a central processing unit (CPU) or processor, a memory, and an interface system which are coupled together by a bus or other link, although each of the web content server devices 16(1)-16(n) could have other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor in each of the web content server devices 16(1)-16(n) executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[00023] The memory in each of the web content server devices 16(1)-16(n) stores these programmed instructions for one or more aspects of the present invention as described and illustrated by way of the embodiments, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor, can be used for the memory in each of the web content server devices 16(1)-16(n).

[00024] The interface system in each of the web content server devices 16(1)-16(n) is used to operatively couple and communicate between the web content server devices 16(1)-16(n) and the proxy server device 12, the mobile computing devices 14(1)-14(n), and the client computing devices 15(1)-15(n) via communication networks 18(1)-18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00025] Although embodiments of the proxy server device 12, the mobile computing devices 14(1)-14(n), and the web content server devices 16(1)-16(n), are described and illustrated herein, each can be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s).

[00026] Furthermore, each of the systems of the embodiments may be conveniently implemented using one or more general purpose computer systems, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[00027] In addition, two or more computing systems or devices can be substituted for any one of the systems in any embodiment of the embodiments. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the embodiments. The embodiments may also be implemented on computer system or systems that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched

Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[00028] The embodiments may also be embodied as non-transitory computer readable medium having instructions stored thereon for one or more aspects of the present invention as described and illustrated by way of the embodiments herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the embodiments, as described and illustrated herein.

[00029] An exemplary method for emailing labels as portable data files and devices thereof will now be described with reference to FIGS. 1-3. Referring more specifically to FIG. 2, in step 100 the proxy server device 12 receives an HTTP request for a web page from one of the mobile computing devices 1491)-14(n). The proxy server device 12 transmits the request to one of the web server devices 16(1)-16(n) hosting the website with the requested content.

[00030] In step 102, the proxy server device 12 receives the requested content, such as a web page including a printable label from the one of the web server devices 16(1)-16(n) hosting the website with the requested content. The printable label can comprise a variety of different printable content which is not viewable on the requesting one of the mobile computing devices 14(1)-14(n), such as a bar code, proof of insurance, medical record, or bank statement by way of example.

[00031] In step 104, the proxy server device 12 determines whether the web page with the printable label from the one of the web server devices 16(1)-16(n) is optimized for printing. If in step 104, the proxy server device 12 determines the web page with the printable label is not optimized for printing, then the No branch is taken to step 106. In step 106, the proxy server device 12 extracts and/or optimizes the printable label for printing and then proceeds to step 108. If in step 104, the proxy server device 12 determines the web page with the printable label is optimized for printing, then the Yes branch is taken to step 108.

[00032] In step 108, the proxy server device 12 determines whether a user at the requesting one of the mobile computing devices 14(1)-14(n) has provided notes and/or logos as well as an email address to receive the generate portable data file, although other types of data and destination information could be used. If in step 108 the proxy server device 12 determines the user at the requesting one of the mobile computing devices 14(1)-14(n) has not provided any notes and/or logos as well as an email address associated with the requesting one of the mobile computing devices 14(1)-14(n), then the No branch is taken to step 110. In step 110, the proxy server device 12 transmits a web page, such as the one shown in FIG. 2, to the requesting one of the mobile computing devices 14(1)-14(n) to obtain notes and/or logos as well as an email address associated with the requesting one of the mobile computing devices 14(1)-14(n) to receive the label, although other manners of obtaining customizing and addressing information relating to the label could be used. The proxy server device 12 adds the obtained notes and/or logos or other data from entries in the fields shown in FIG. 2 to customize the label and then proceeds to step 112.

[00033] If in step 108 the proxy server device 12 determines the user at the requesting one of the mobile computing devices 14(1)-14(n) has not added notes and/or logos for the label and already has provided an email address, then the Yes branch is taken to step 112. In step 112, the proxy server device 12 converts the customized label into a PDF file, although the label can be converted in other numbers and types of portable data files for transmission.

[00034] In step 114, the proxy server device 12 transmits the generated label to the email address associated with the requesting one of the mobile computing devices 14(1)-14(n). The proxy server device 12 may optionally send a confirmation to the requesting one of the mobile computing devices 14(1)-14(n) that the PDF or other portable data file has been sent. The user associated with the requesting one of the mobile computing devices 14(1)-14(n) can later access the email account through for example a desktop computing device connected to a printer to print the customized label which was not viewable on the requesting one of the mobile computing devices 14(1)-14(n).

[00035] Accordingly, as illustrated and described herein this technology provides a number of advantages including providing a method, computer readable medium and an apparatus that enables a mobile computing device to create a portable data file of one or more labels and send the portable data file, along with custom information, such as descriptions and logos, to a designated email address associated with the mobile computing device. One of the advantages of this technology is that mobile computing devices can now obtain labels that were not viewable on the mobile computing device.

[00036] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for emailing a label as a portable data file to destination information associated with a mobile computing device, the method comprising:

obtaining at a proxy server device a web page with a label from a web server device in response to a request from a mobile computing device;

obtaining with the proxy server device destination information for the label associated with the requesting mobile computing device;

converting with the proxy server device the label into a portable data file; and

transmitting with the proxy server device the portable data file using the destination information for the label associated with the requesting mobile computing device.

2. The method as set forth in claim 1 further comprising:

determining with the proxy server device whether the web page with the label is optimized for printing; and

extracting and optimizing with the proxy server device the label for printing when the determining indicates the web page and label are not optimized for printing.

3. The method as set forth in claim 1 further comprising:

determining with the proxy server device whether the requesting mobile computing device has provided at least one of notes and a logo to customize the label; and

customzing with the proxy server device the label when the requesting mobile computing device has provided at least one of the notes and the logo.

4. The method as set forth in claim 1 wherein the portable data file is a PDF file.

5. The method as set forth in claim 1 wherein the obtained destination information comprises an email address.

6. A non-transitory computer readable medium having stored thereon instructions for emailing a label as a portable data file to destination information associated with a mobile computing device comprising machine executable code which when executed by at least one processor, causes the processor to perform steps comprising:

obtaining a web page with a label from a web server device in response to a request from a mobile computing device;

obtaining destination information for the label associated with the requesting mobile computing device;

converting the label into a portable data file; and

transmitting the portable data file using the destination information for the label associated with the requesting mobile computing device.

7. The medium as set forth in claim 6 further comprising:

determining whether the web page with the label is optimized for printing; and

extracting and optimizing the label for printing when the determining indicates the web page and label are not optimized for printing.

8. The medium as set forth in claim 6 further comprising:

determining whether the requesting mobile computing device has provided at least one of notes and a logo to customize the label; and

customzing the label when the requesting mobile computing device has provided at least one of the notes and the logo.

9. The medium as set forth in claim 6 wherein the portable data file is a PDF file.

10. The medium as set forth in claim 6 wherein the obtained destination information comprises an email address.

11. A proxy server device comprising:

one or more processors;

a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory comprising:

obtaining a web page with a label from a web server device in response to a request from a mobile computing device;

obtaining destination information for the label associated with the requesting mobile computing device;

converting the label into a portable data file; and

transmitting the portable data file using the destination information for the label associated with the requesting mobile computing device.

12. The device as set forth in claim 11 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising:

determining whether the web page with the label is optimized for printing; and

extracting and optimizing the label for printing when the determining indicates the web page and label are not optimized for printing.

13. The device as set forth in claim 11 wherein the one or more processors is further configured to execute programmed instructions stored in the memory further comprising:

determining whether the requesting mobile computing device has provided at least one of notes and a logo to customize the label; and

customzing the label when the requesting mobile computing device has provided at least one of the notes and the logo.

14. The device as set forth in claim 11 wherein the portable data file is a PDF file.

15. The device as set forth in claim 11 wherein the obtained destination information comprises an email address.

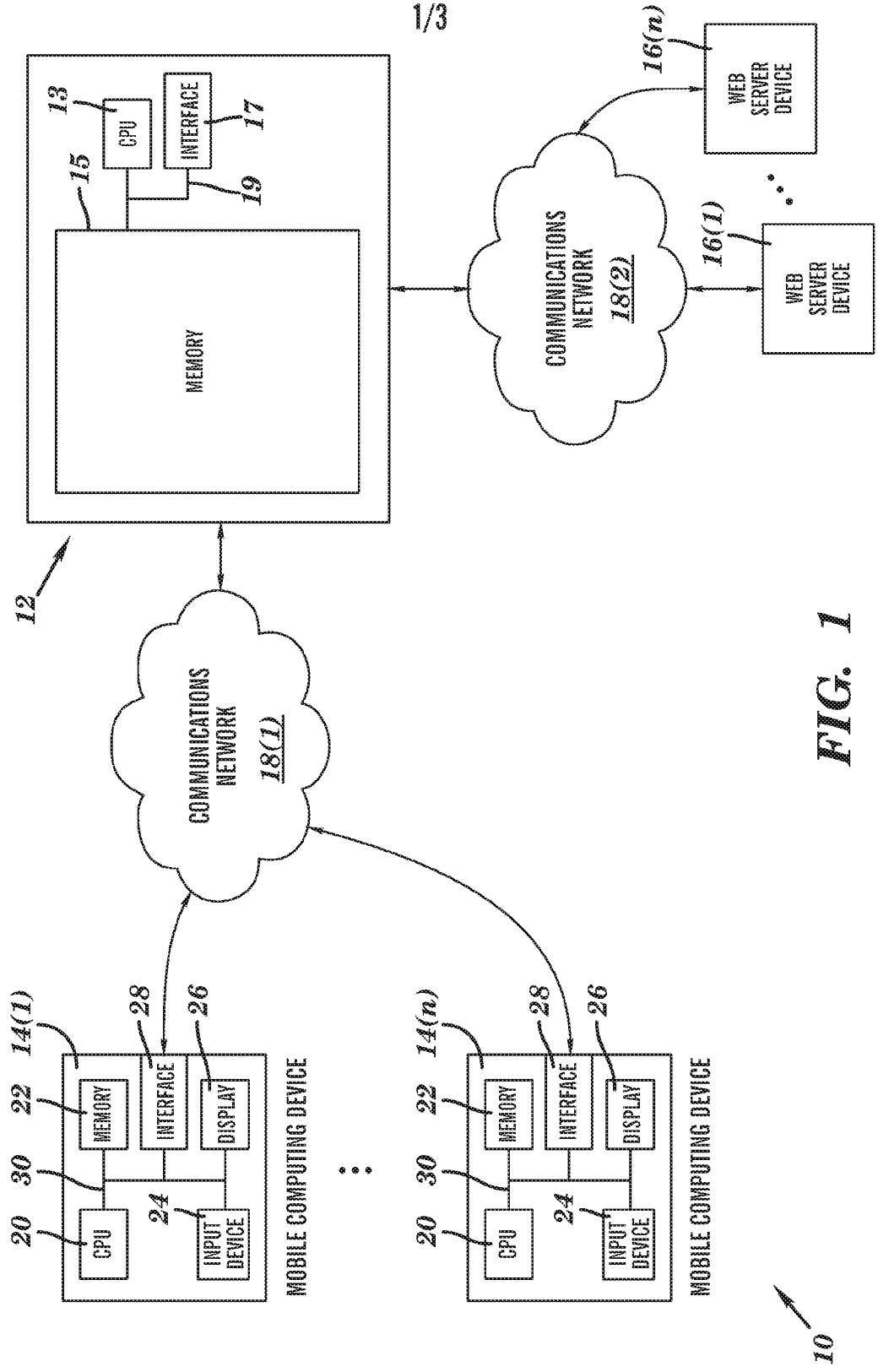

12
13
CPU
INTERFACE
17
15
19
MEMORY
COMMUNICATIONS NETWORK 18(2)
16(n)
WEB SERVER DEVICE
16(1)
WEB SERVER DEVICE
COMMUNICATIONS NETWORK 18(1)
14(1)
28
26
22
MEMORY
INTERFACE
DISPLAY
30
20
CPU
24
INPUT DEVICE
MOBILE COMPUTING DEVICE
14(n)
28
26
22
MEMORY
INTERFACE
DISPLAY
30
20
CPU
24
INPUT DEVICE
MOBILE COMPUTING DEVICE
10

FIG. 1

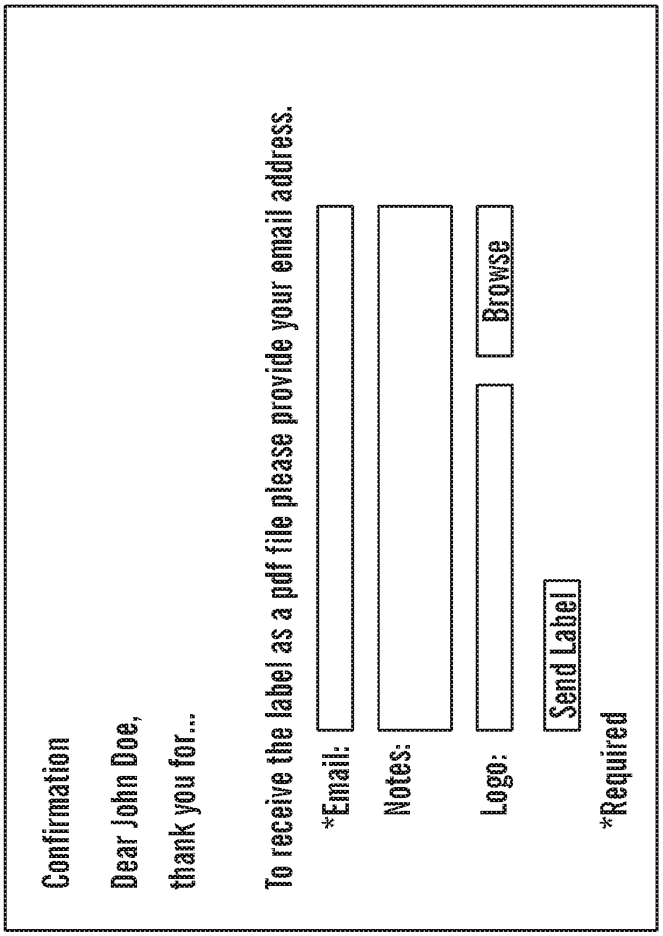

Confirmation
Dear John Doe,
thank you for...
To receive the label as a pdf file please provide your email address.
*Email:
Notes:
Logo:
Browse
Send Label
*Required

FIG. 2

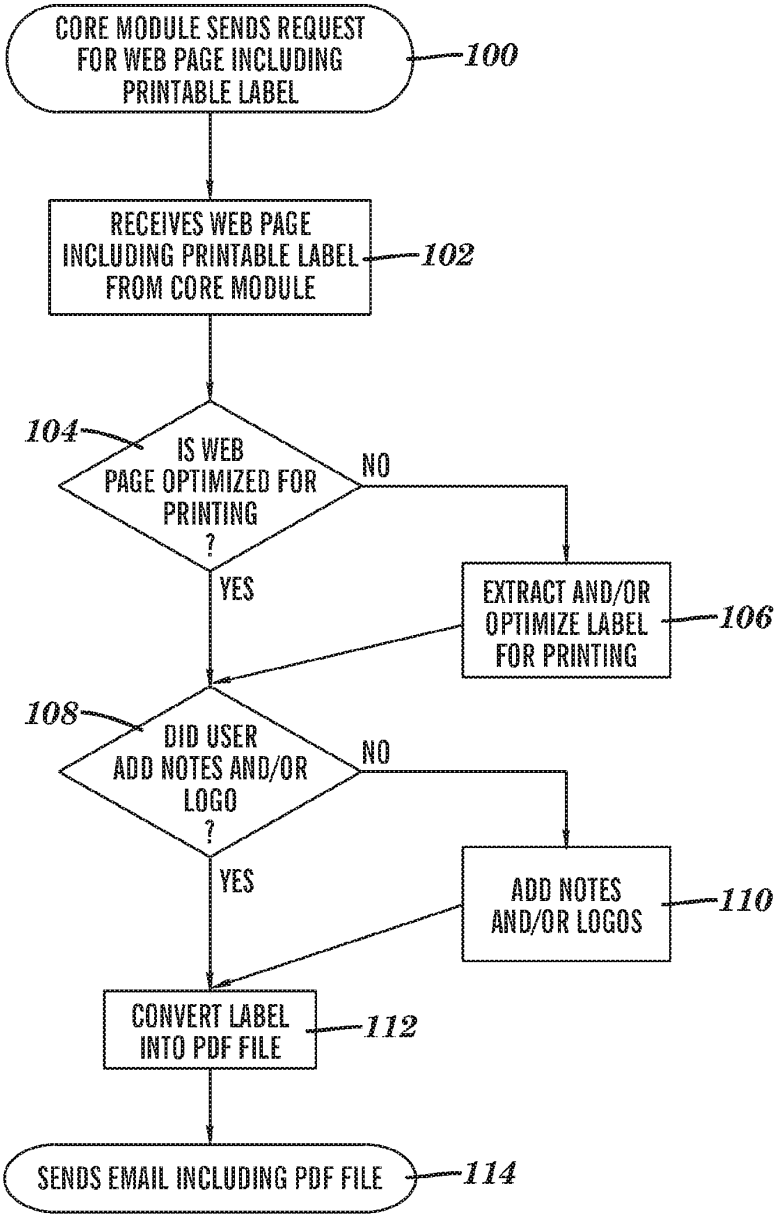

CORE MODULE SENDS REQUEST FOR WEB PAGE INCLUDING PRINTABLE LABEL
100
RECEIVES WEB PAGE INCLUDING PRINTABLE LABEL FROM CORE MODULE
102
104
IS WEB PAGE OPTIMIZED FOR PRINTING ?
NO
YES
EXTRACT AND/OR OPTIMIZE LABEL FOR PRINTING
106
108
DID USER ADD NOTES AND/OR LOGO ?
NO
YES
ADD NOTES AND/OR LOGOS
110
CONVERT LABEL INTO PDF FILE
112
SENDS EMAIL INCLUDING PDF FILE
114

FIG. 3

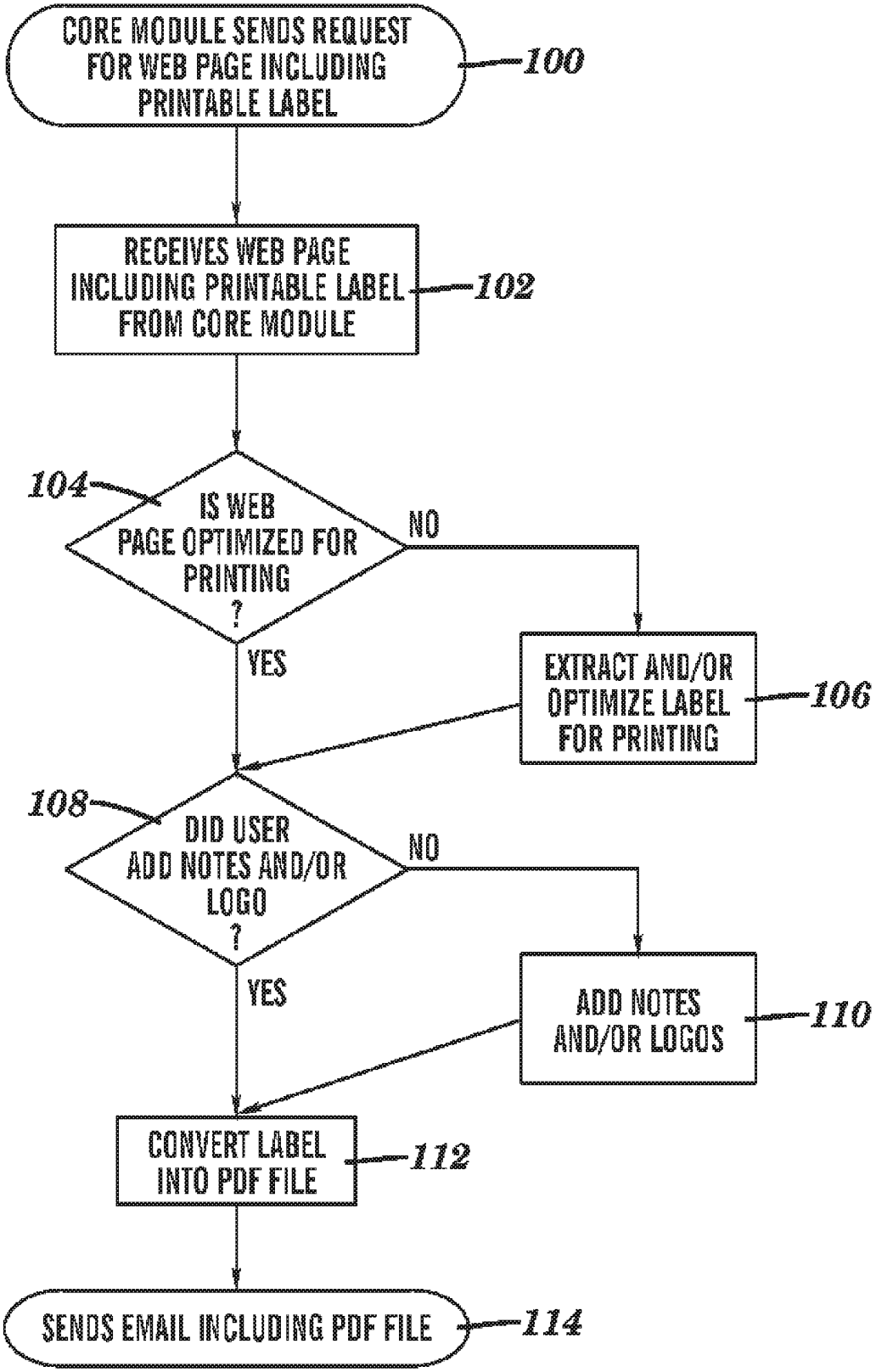

CORE MODULE SENDS REQUEST FOR WEB PAGE INCLUDING PRINTABLE LABEL
100
RECEIVES WEB PAGE INCLUDING PRINTABLE LABEL FROM CORE MODULE
102
104
IS WEB PAGE OPTIMIZED FOR PRINTING ?
NO
YES
EXTRACT AND/OR OPTIMIZE LABEL FOR PRINTING
106
108
DID USER ADD NOTES AND/OR LOGO ?
NO
YES
ADD NOTES AND/OR LOGOS
110
CONVERT LABEL INTO PDF FILE
112
SENDS EMAIL INCLUDING PDF FILE
114

FIG. 3

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2824861 A1 2014/03/10

(21) 2 824 861

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(22) Date de dépôt/Filing Date: 2013/08/28

(41) Mise à la disp. pub./Open to Public Insp.: 2014/03/10

(30) Priorités/Priorities: 2012/09/10 (US61/698,948); 2013/03/12 (US13/796,857)

(51) Cl.Int./Int.Cl. *H04L 12/16* (2006.01), *G06F 17/00* (2006.01), *G06Q 30/00* (2012.01)

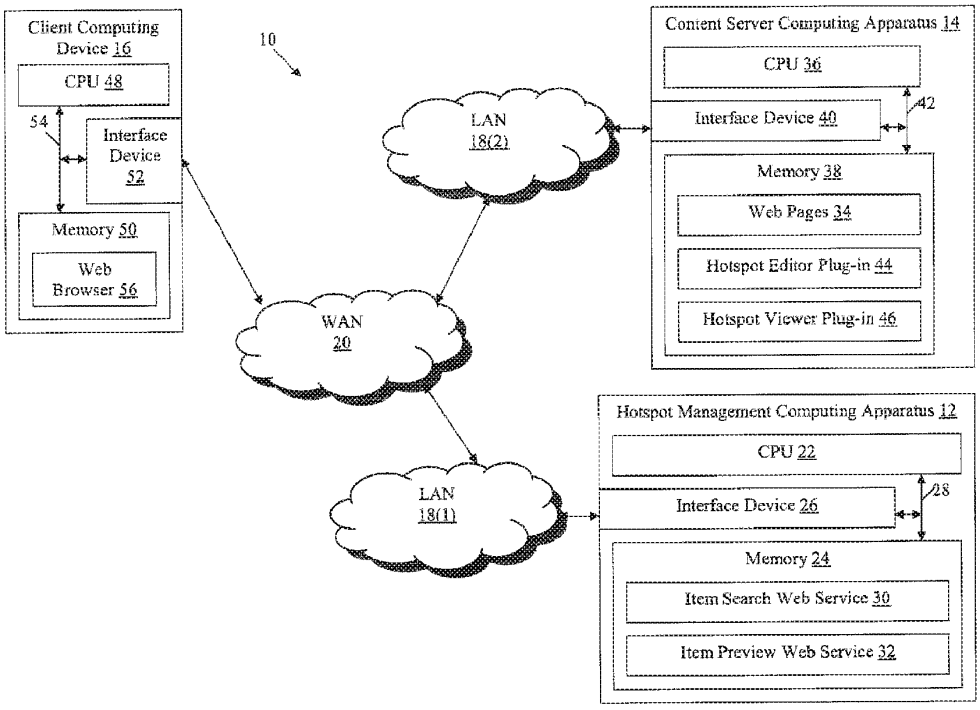
(71) Demandeur/Applicant:
USABLENET INC., US

(72) Inventeur/Inventor:
SCODA, ENRICO, IT

(74) Agent: GOWLING LAFLEUR HENDERSON LLP

(54) Titre : METHODES EN VUE DE FACILITER LE REPERAGE DE ZONES SENSIBLES DE PAGE WEB ET SES DISPOSITIFS

(54) Title: METHODS FOR FACILITATING WEB PAGE IMAGE HOTSPOTS AND DEVICES THEREOF



(57) Abrégé/Abstract:
A method, non-transitory computer readable medium, and hotspot management computing apparatus that receives a JavaScript Object Notation with Padding (JSONP) request from a client computing device, wherein the JSONP request comprises an identifier associated with a selected hotspot associated with an image of a first web page. A HyperText Markup Language (HTML) fragment is generated based on the identifier, wherein the HTML fragment defines a first description associated with the hotspot and an input element associated with a Uniform Resource Locator (URL) of a second web page including a second description associated with the hotspot. The HTML fragment is sent to the requesting client computing device.

ABSTRACT

A method, non-transitory computer readable medium, and hotspot management computing apparatus that receives a JavaScript Object Notation with Padding (JSONP) request from a client computing device, wherein the JSONP request comprises an identifier associated with a selected hotspot associated with an image of a first web page. A HyperText Markup Language (HTML) fragment is generated based on the identifier, wherein the HTML fragment defines a first description associated with the hotspot and an input element associated with a Uniform Resource Locator (URL) of a second web page including a second description associated with the hotspot. The HTML fragment is sent to the requesting client computing device.

METHODS FOR FACILITATING WEB PAGE IMAGE HOTSPOTS AND DEVICES THEREOF

[0001] This application claims the benefit of U.S. Provisional Patent Application Serial No. 61/698,948, filed September 10, 2012, which is hereby incorporated by reference in its entirety.

FIELD

[0002] This technology generally relates to web page image hotspots and, more particularly, to methods, non-transitory computer readable medium, apparatuses that facilitate establishing and interacting with web page image hotspots.

BACKGROUND

[0003] Images have become increasingly pervasive across the Internet, particularly with respect to social media and retail web sites for example, although many different types of web sites host images. Hotspots, or identified regions of images, are often established in order for users to associate metadata or other information with the images. For example, a host of an image on a social media web site can establish a hotspot (often referred to as "tagging") to indicate another user present in the image. Upon selection of the hotspot, the other user's name and a link to the other user's profile can be displayed, for example. On a retail web site, a host may identify a region of an image including an item for sale as a hotspot. Upon selection of the hotspot by a user, content regarding the item can be displayed, often with a link to a more detailed description of the item.

[0004] However, web browsers generally prohibit communication across domains based on a same origin policy. Accordingly, the available content for a user to associate with a web page image hotspot is generally limited to content hosted by a web server in the same domain as the web page that includes the image. Additionally, web services hosted by web servers, and configured to identify content to be associated with a hotspot, are often developed for a single domain. Accordingly, such web services do not provide an interface

or structure common to any other web service configured to provide similar functionality but associated with a different domain.

SUMMARY

[0005] A method for facilitating interaction with a web page image hotspot includes receiving, by a hotspot management computing apparatus, a JavaScript Object Notation with Padding (JSONP) request from a client computing device, wherein the JSONP request comprises an identifier associated with a selected hotspot associated with an image of a first web page. A HyperText Markup Language (HTML) fragment is generated, by the hotspot management computing apparatus, based on the identifier, wherein the HTML fragment defines a first description associated with the hotspot and an input element associated with a Uniform Resource Locator (URL) of a second web page including a second description associated with the hotspot. The HTML fragment is sent, by the hotspot management computing apparatus, to the requesting client computing device.

[0006] A computer readable medium having stored thereon instructions for facilitating interaction with a web page image hotspot comprising machine executable code which when executed by a processor, causes the processor to perform steps including receiving a JSONP request from a client computing device, wherein the JSONP request comprises an identifier associated with a selected hotspot associated with an image of a first web page. An HTML fragment is generated based on the identifier, wherein the HTML fragment defines a first description associated with the hotspot and an input element associated with a URL of a second web page including a second description associated with the hotspot. The HTML fragment is sent to the requesting client computing device.

[0007] A hotspot management computing apparatus includes a memory coupled to a processor which is configured to execute programmed instructions stored in the memory comprising receiving a JSONP request from a client computing device, wherein the JSONP request comprises an identifier associated with a selected hotspot associated with an image of a first web page. An HTML fragment is generated based on the identifier, wherein the HTML fragment defines a first description associated with the hotspot and an input element

associated with a URL of a second web page including a second description associated with the hotspot. The HTML fragment is sent to the requesting client computing device.

[0008] A method for generating a web page image hotspot includes receiving, with a hotspot management computing apparatus, a JSONP request comprising search criteria. Content satisfying the search criteria is identified by the hotspot management computing apparatus. An HTML fragment defining an input element associated with each of one or more search results and an identifier is generated by the hotspot management computing apparatus, wherein each of the one or more search results includes at least a portion of the identified content. The HTML fragment is sent to the requesting client computing device by the hotspot management computing apparatus.

[0009] A computer readable medium having stored thereon instructions for facilitating generation of a web page image hotspot comprising machine executable code which when executed by a processor, causes the processor to perform steps including receiving a JSONP request comprising search criteria. Content satisfying the search criteria is identified. An HTML fragment defining an input element associated with each of one or more search results and an identifier is generated, wherein each of the one or more search results includes at least a portion of the identified content. The HTML fragment is sent to the requesting client computing device.

[0010] A hotspot management computing apparatus includes a memory coupled to one or more processors which are configured to execute programmed instructions stored in the memory including receiving a JSONP request comprising search criteria. Content satisfying the search criteria is identified. An HTML fragment defining an input element associated with each of one or more search results and an identifier is generated, wherein each of the one or more search results includes at least a portion of the identified content. The HTML fragment is sent to the requesting client computing device.

[0011] A method for generating a web page image hotspot includes generating, by a client computing device, a search panel in response to a user identifying a region of an image rendered on a web page as a hotspot, the search panel comprising one or more input fields. A

JSONP request is sent, by the client computing device, based on interaction with the input element of the search panel, to a hotspot management computing apparatus, wherein the JSONP request comprises search criteria entered in the one or more input fields. An HTML fragment received from the hotspot management computing apparatus is rendered, by the client computing device, in a results panel, the HTML fragment defining an input element associated with each of one or more search results, wherein each input element is associated with an identifier of one of the search results. The identifier associated with a selected one of the input elements is sent, by the client computing device, to the hotspot management computing apparatus.

[0012] A non-transitory computer readable medium having stored thereon instructions for generating a web page image hotspot comprising machine executable code which when executed by a processor, causes the processor to perform steps including generating a search panel in response to a user identifying a region of an image rendered on a web page as a hotspot, the search panel comprising one or more input fields. A JSONP request is sent based on interaction with the input element of the search panel, to a hotspot management computing apparatus, wherein the JSONP request comprises search criteria entered in the one or more input fields. An HTML fragment received from the hotspot management computing apparatus is rendered in a results panel, the HTML fragment defining an input element associated with each of one or more search results, wherein each input element is associated with an identifier of one of the search results. The identifier associated with a selected one of the input elements is sent to the hotspot management computing apparatus.

[0013] A client computing device includes a memory coupled to a processor which is configured to execute programmed instructions stored in the memory including generating a search panel in response to a user identifying a region of an image rendered on a web page as a hotspot, the search panel comprising one or more input fields. A JSONP request is sent based on interaction with the input element of the search panel, to a hotspot management computing apparatus, wherein the JSONP request comprises search criteria entered in the one or more input fields. An HTML fragment received from the hotspot management computing

apparatus is rendered in a results panel, the HTML fragment defining an input element associated with each of one or more search results, wherein each input element is associated with an identifier of one of the search results. The identifier associated with a selected one of the input elements is sent to the hotspot management computing apparatus.

[0014] A method for interacting with a web page image hotspot includes displaying, with a client computing device, one or more hotspots of an image included on a first web page. A JSONP request is sent, with the client computing device, to a hotspot management computing apparatus in response to user interaction with one of the one or more hotspots, wherein the JSONP request comprises an identifier associated with the one hotspot. An HTML fragment received from the hotspot management computing apparatus is rendered, with the client computing device, in a preview panel, wherein the HTML fragment is based on the identifier and defines a first description associated with the one hotspot and an input element associated with a URL of a second web page including a second description associated with the one hotspot. The second web page is displayed, with the client computing device, in response to user interaction with the input element of the preview panel.

[0015] A non-transitory computer readable medium having stored thereon instructions for interacting with a web page image hotspot comprising machine executable code which when executed by a processor, causes the processor to perform steps including displaying one or more hotspots of an image included on a first web page. A JSONP request is sent, with the client computing device, to a hotspot management computing apparatus in response to user interaction with one of the one or more hotspots, wherein the JSONP request comprises an identifier associated with the one hotspot. An HTML fragment received from the hotspot management computing apparatus is rendered, with the client computing device, in a preview panel, wherein the HTML fragment is based on the identifier and defines a first description associated with the one hotspot and an input element associated with a URL of a second web page including a second description associated with the one hotspot. The second web page is displayed, with the client computing device, in response to user interaction with the input element of the preview panel.

[0016] A client computing device including a memory coupled to a processor which is configured to execute programmed instructions stored in the memory including displaying one or more hotspots of an image included on a first web page. A JSONP request is sent, with the client computing device, to a hotspot management computing apparatus in response to user interaction with one of the one or more hotspots, wherein the JSONP request comprises an identifier associated with the one hotspot. An HTML fragment received from the hotspot management computing apparatus is rendered, with the client computing device, in a preview panel, wherein the HTML fragment is based on the identifier and defines a first description associated with the one hotspot and an input element associated with a URL of a second web page including a second description associated with the one hotspot. The second web page is displayed, with the client computing device, in response to user interaction with the input element of the preview panel.

[0017] This technology provides a number of advantages including methods, non-transitory computer readable medium, and apparatuses that facilitate establishing web page image hotspots using a plug-in sent along with a web page. The plug-in can obtain data to be associated with the hotspot from a web service hosted by a different domain using JSONP. This technology also facilitates user interaction with web page image hotspots using a plug-in configured to communicate using JSONP with a web service, which can also be hosted by a different domain.

BRIEF DESCRIPTION OF THE DRAWINGS

[0018] FIG. 1 is a block diagram of a network environment which incorporates an exemplary hotspot management computing apparatus;

[0019] FIG. 2 is a flowchart of an exemplary method of inserting a web page image hotspot;

[0020] FIG. 3 is a flowchart of ;an exemplary method of facilitating the association of a web page image hotspot with content;

[0021] FIG. 4 is a flowchart of an exemplary method of displaying a web page image hotspot;

[0022] FIG. 5 is a flowchart of an exemplary method of facilitating the display of a web page image hotspot;

[0023] FIG. 6 is an exemplary web page including an image and a plurality of hotspots associated with the image;

[0024] FIG. 7 is an exemplary search panel for obtaining search criteria for content to be associated with a web page image hotspot;

[0025] FIG. 8 is an exemplary search results panel including identified content to be associated with a web page image hotspot; and

[0026] FIG. 9 is an exemplary web page including a plurality of hotspots and an exemplary preview panel associated with one of the hotspots.

DETAILED DESCRIPTION

[0027] An exemplary network environment 10 is illustrated in FIG. 1 as including an exemplary hotspot management computing apparatus 12. In this example, the hotspot management computing apparatus 12 is coupled to a content server computing apparatus 14 and a client computing device 16 by local area networks (LANs) 18(1) and 18(2) and a wide area network (WAN) 20, although other types and numbers of devices, components, and elements in other topologies could be used. This technology provides a number of advantages including methods, non-transitory computer readable medium, and apparatuses for more easily and effectively generating and viewing web page image hotspots across domains and sources of content associated with the hotspots.

[0028] Referring more specifically to FIG. 1, the hotspot management computing apparatus 12 includes at least one processor or CPU 22, a memory 24, and an interface device 26, which are coupled together by a bus 28 or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and

locations can be used. The processor 22 of the hotspot management computing apparatus 12 may execute one or more stored programmed instructions for one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor 22 could execute other numbers and types of programmed instructions.

[0029] The memory 24 of the hotpot management computing apparatus 12 stores these programmed instructions for one or more aspects of the present invention, as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. The memory 24 of the hotspot management computing apparatus 12 may include one or more tangible storage media and/or devices, such as RAM, ROM, flash memory, CD-ROM, floppy disk, hard disk drive(s), solid state memory, DVD, or any other memory storage types or devices, including combinations thereof, which are known to those of ordinary skill in the art.

[0030] In this example, the memory 24 of the hotpot management computing apparatus 12 includes an item search web service 30 and an item preview web service 32 including JavaScript instructions, for example, for one or more aspects of the present invention as described and illustrated herein, although the memory 24 can include other types and numbers of systems, devices, and elements in other configurations.

[0031] The interface device 26 in the hotspot management computing apparatus 12 is used to operatively couple and communicate between the hotspot management computing apparatus 12 and the client computing device 16 via LAN 18(1) and WAN 20, although other types and numbers of communication networks or systems with other types and numbers of connections and configurations to other devices and elements can also be used. The LANs 18(1) and 18(2) and WAN 20 can use TCP/IP over Ethernet and industry-standard protocols, including NFS, CIFS, SOAP, XML, LDAP, and SNMP, for example, although other types and numbers of communication networks can also be used.

[0032] Generally, the content server computing apparatus 14 processes requests for web pages 34 received from the client computing device 16 via LAN 18(2) and WAN 20 according to the HTTP-based protocol, for example. The content server computing apparatus

14 includes at least one processor or CPU 36, a memory 38, and an interface device 40, which are coupled together by a bus 42 or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can also be used. The processor 36 in the content server computing apparatus 14 executes a program of stored instructions for one or more aspects of the present invention, as described and illustrated by way of the embodiments herein, although the processor 36 could execute other numbers and types of programmed instructions.

[0033] The memory 38 in the content server computing apparatus 14 stores these programmed instructions for one or more aspects of the present invention, as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other non-transitory computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor 36, can be used for the memory 38 in the content server computing apparatus 14.

[0034] In this example, the memory 38 includes a hotspot editor plug-in 44 and a hotspot viewer plug-in 46 including JavaScript instructions, for example, configured to be included in any of the web pages stored by the memory and provided in response to a client computing device request. The memory 38 can further include other types and numbers of systems, devices, and elements in other configurations which store other data.

[0035] The interface device 40 in the content server computing apparatus 14 is used to operatively couple and communicate between the content server computing apparatus 14 and the client computing device 16 via LAN 18(2) and WAN 20, although other types and numbers of communication networks with other types and numbers of connections and configurations can be used. The content server computing apparatus 14 may be hardware or software or may represent a system with multiple content server computing apparatuses in a pool, which may include internal or external networks. In this example the content server

computing apparatus 14 may be any version of Microsoft® IIS server or Apache® server, although other types of content server computing apparatus 14 may be used.

[0036] The client computing device 16 enables a user to request, receive and interact with services and content hosted by the content server computing apparatus 14 and hotspot management computing apparatus 12 via the LANs 18(1) and 18(2) and WAN 20, although the client computing device 16 could access content and utilize other types and numbers of content or applications from other sources and could provide a wide variety of other functions for a user. By way of example only, the client computing device 16 can be a mobile computing device, smart phone, personal digital assistant, or computer, for example.

[0037] In this example, the client computing device 16 includes at least one processor or CPU 48, a memory 50, and an interface device 52, and which are coupled together by a bus 54 or other link, although the client computing device 16 can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 48 in the client computing device 16 executes a program of stored instructions for one or more aspects of the present invention as described and illustrated herein, although the processor 48 could execute other numbers and types of programmed instructions.

[0038] The memory 50 in the client computing device 16 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a RAM or a ROM in the system or a floppy disk, hard disk, CD ROM, or other non-transitory computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor 48 can be used for the memory 50 in the client computing device 16. In this example, the client computing device 16 is configured to access web services and web content through a web browser 56 stored in the memory 50.

[0039] The interface device 52 in the client computing device 16 is used to operatively couple and communicate between the client computing device 16 and the content

server computing apparatus 14 and the hotspot management computing apparatus 12 via the LANs 18(1) and 18(2) and the WAN 20, although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[0040] Although embodiments of the hotspot management computing apparatus 12, content server computing apparatus 14, and client computing device 16 are described and illustrated herein, each of these devices can be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s). Furthermore, each of the systems of the embodiments may be conveniently implemented using one or more general purpose computer systems, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[0041] In addition, two or more computing systems or devices can be substituted for any one of the devices in any embodiment. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the embodiments. The embodiments may also be implemented on computer system(s) that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[0042] The examples may also be embodied as a non-transitory computer readable medium having instructions stored thereon for one or more aspects of the present technology as described and illustrated by way of the examples herein, as described herein, which when

executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the examples, as described and illustrated herein.

[0043] An exemplary method for facilitating web page image hotspots will now be described with reference to FIGS. 1-9. Referring specifically to FIG. 2, an exemplary method of inserting a web page image hotspot will now be described. In this example in step 200, a user of the client computing device 16 requests and obtains, using the web browser 56, one of the web pages 34 hosted by the content server computing apparatus 14 and including one or more images. An exemplary one of the web pages 34 stored in the memory 24 of the content server computing apparatus 14 is illustrated in FIG. 6 as web page 600, which includes an image 602.

[0044] In this example, along with sending the requested web page 600 to the client computing device 16, the content server computing apparatus 14 sends the hotspot editor plug-in 44. The content server computing apparatus 14 can send the hotspot editor plug-in 44 by inserting the JavaScript code of the plug-in 44 into the source code of the requested web page 600. In another example, the content server computing apparatus 14 provides the hotspot editor plug-in 44 along with the requested web page 600 as a bookmarklet. In yet another example, the hotspot editor plug-in 44 can be integrated within a platform, such as a blog engine or a social network platform, for example. Other methods of providing the hotspot editor plug-in 44 can also be used. Upon receiving the requested web page 600, the web browser 52 of the client computing device 16 attempts to render the requested web page on a display of the client computing device 16.

[0045] In step 202, while rendering the web page 600, and upon encountering the JavaScript code of the hotspot editor plug-in 44, the web browser 52 of the client computing device 16 executes the code to generate a hotspot identification interface. The interface is configured to facilitate identifying a region of the image 602 of the requested web page 600 as a hotspot. The interface can be presented visually by the web browser 52, such as in an overlay, new panel, or new window, for example, which allows selection of the hotspot features (e.g. size or representation style) by the user. Alternatively, the interface can be run

in the background without a graphical interface while still allowing the user to select regions of the image 602 of the web page 600 to be identified as hotspots.

[0046] In step 204, the web browser 52 of the client computing device 16 executing the hotspot editor plug-in 44 generates a search panel. In this example, the web browser 52 generates the search panel in response to a user of the client computing device 16 identifying a region of the image 602 of the requested web page 600 as a hotspot using the interface generated in step 202. An exemplary identified hotspot of the image 602 of the web page 600 is illustrated as hotspot 604 in FIG. 6. Referring to FIG. 7, an exemplary search panel 700 generated by the web browser 52 of the client computing device 16 in response to the user identifying a region of the image 602 as hotspot 604 is illustrated. The search panel 700 includes one or more input fields, such as the search text field 702, a source selector element 704, and an input element, such as the search button 706, although other types and number of input fields and elements can also be used.

[0047] In this example, the source selector element 704 is a drop-down menu identifying a plurality of optional sources selectable by the user. Each source is associated by the hotspot editor plug-in 44 with a Uniform Resource Locator (URL) of an item search web service, such as the item search web service 30 stored in the memory 24 of the hotspot management computing apparatus 12, as described and illustrated in more detail below. The source selector element 704 can be populated with sources, and associated URLs, based on an identifier associated with the user sent to the content server computing apparatus 12 during an initialization or registration process. In another example, the source selector element 704 can be populated with sources, and associated URLs, based on a hardcoding by an administrator or developer of the hotspot editor plug-in 44. Other methods for populating the source selector element can also be used.

[0048] In step 206, the web browser 52 of the client computing device 16 executing the hotspot editor plug-in 44 sends, upon user interaction with the input element 706 of the search panel 700, a JavaScript Object Notation with Padding (JSONP) request to the item search web service 30 of the hotspot management computing apparatus 12. In this example, the JSONP request includes search criteria input using the input field 702 of the search panel

700 and is sent to the item search web service 30 located at the URL associated with the source selected by the user in the source selector element 704. Exemplary sources identified using the source selector element can be a product catalog, a blog post, or a social network feed, although any other types of sources can also be used.

[0049] Each source and/or item search web service 30 can be located on the same or different hotspot management computing apparatus 12. The search criteria input by the user is used by the item search web service 30 to locate content that can be presented to and identified by the user as associated with the hotspot 604. Optionally, the JSONP request further includes an attribute indicating a layout to be used as described and illustrated in more detail below. The layout can be a default layout included in the JavaScript code of the hotspot editor plug-in 44, for example, although other methods of indicating a layout can also be used.

[0050] In this example, the JSONP request is sent to a URL endpoint associated with the selected source. Additionally, the JSONP request identifies the item search web service 30 hosted by a hotspot management computing apparatus 12. By using the JSONP protocol, the hotspot editor plug-in 44 can advantageously communicate with an item search web service 30 that is cross-domain or located at a URL including a different domain, host, port, and/or application layer protocol than a URL of the content server computing apparatus 14 storing the web page 600 and the hotspot editor plug-in 44.

[0051] Accordingly, with this example of the technology, the hotspot editor plug-in 44 and item search web service 30 are decoupled such that the plug-in 44 can be included with or inserted into the source code of any of the web pages 34 and the item search web service 30 can be located on any server computing device despite web browser enforcement of a same origin policy. The URL of the item search web service 30 can be included in the hotspot editor plug-in 44 such that user interaction with the input element 706 of the search panel 700 initiates the JSONP request to the item search web service 30 located at the URL.

[0052] In step 208, the web browser 52 of the client computing device 16 executing the hotspot editor plug-in 44 renders a HyperText Markup Language (HTML) fragment. The

HTML fragment is received by the web browser 52 of the client computing device from, and is generated by, the item search web service 30, as described in more detail below. The HTML fragment, when rendered, is configured to display a results panel 800, such as the results panel 800 illustrated in FIG. 8, for example. The content of the results panel 800 can also be output to a display of the client computing device 16 in the form of a new window or overlay, for example, although any other form of display can also be used.

[0053] Optionally, the item search web service 30 can generate the HTML fragment according to a layout attribute value received in the JSONP request sent by the client computing device 16 in step 206. Referring more specifically to FIG. 8, in this example, the HTML fragment defines results panel input elements 802(1) and 802(2), each of which is associated with one of the search results 804(1) and 804(2). Additionally, each of the results panel input elements 802(1) and 802(2) is associated with an attribute having a value of an identifier of an associated one of the search results 804(1) and 804(2). Accordingly, in this example, the item search web service 30 processes the search criteria sent in the JSONP request in step 206 and sends a response with search results 804(1) and 804(2) and including an HTML fragment defining the display of those search results 804(1) and 804(2).

[0054] Referring back to FIG. 2, in step 210, the client computing device 16 executing the hotspot editor plug-in 44 stores the identifier associated with one of the results panel input elements 802(1) and 802(2) selected by a user to be associated with the identified hotspot 604. For example, the hotspot editor plug-in 400 can be configured to send the association information to the hotspot management computing apparatus 12 for storage in the memory 24. Thereby, the hotspot 604 of the image 602 of the web page 600 will be associated with the content of one of the search results 804(1) and 804(2) corresponding to the selected one of the results panel input elements 802(1) and 802(2). Accordingly, the content of the one of the search results 804(1) and 804(2) can be displayed by a web browser of another client computing device upon subsequent retrieval of the web page 600 and selection of the hotspot 604 by a user of the other client computing device, as described and illustrated in more detail below.

[0055] Referring more specifically to FIG. 3, an exemplary method of facilitating the association of a web page image hotspot with content is illustrated. In this example, in step 300, the hotspot management computing apparatus 12 implementing the item search web service 30 receives a JSONP request from a hotspot editor plug-in 44, such as the JSONP request sent in step 206. As discussed above, the item search web service 30 of the hotspot management computing apparatus 12 can be located at a different domain, host, port, and/or application layer protocol than a URL of the content server computing apparatus 14 that provided the hotspot editor plug-in 44 that originated the JSONP request. The JSONP request includes search criteria submitted by a user of the client computing device 16 using the search results panel 700, such as a source of the content to be associated with the hotspot and text search terms for the content.

[0056] In step 302, the hotspot management computing apparatus 12 implementing the item search web service 30 identifies content satisfying the search criteria to generate search result(s). Accordingly, the source and text inputs, for example, can be used by the item search web service 30 to locate possible content that the user of the client computing device 16 may want to associate with an identified hotspot, such as hotspot 604 of the image 602 on the web page 600, for example, as illustrated in FIG. 6.

[0057] In step 304, the hotspot management computing apparatus 12 implementing the item search web service 30 generates an HTML fragment defining an input element associated with each search result, such as input elements 802(1) and 802(2) associated with search result 804(1) and 804(2), respectively, illustrated in FIG. 8. Additionally, an identifier of each search result 804(1) and 804(2) is associated by the item search web service 30 with each input element 802(1) and 802(2) by the HTML fragment. Optionally, the JSONP request further includes an indication of a layout to be used for presenting the search results 804(1) and 804(2) and the HTML fragment can be generated according to the layout.

[0058] In step 306, the hotspot management computing apparatus 12 implementing the item search web service 30 sends the HTML fragment to the requesting hotspot editor plug-in 44 executed by the web browser 52 of the client computing device 16. The results

panel 800 shown in FIG. 8 is an example of a rendering by the web browser 52 of the client computing device 16 of the HTML fragment generated in step 306.

[0059] Referring more specifically to FIG. 4, an exemplary method of displaying a web page image hotspot is shown. In this example, in step 400, a user of the client computing device 16 requests and obtains, using the web browser 52, one of the web pages 34 hosted by the content server computing apparatus 14 and including one or more images, such as the exemplary web page 600 illustrated in FIG. 6 which includes the image 602. Along with sending the requested web page 600 to the client computing device 16, the content server computing apparatus 14 sends the hotspot viewer plug-in 46, such as by inserting the JavaScript code of the hotspot viewer plug-in 46 into the source code of the requested web page 600, although other methods of providing the hotspot viewer plug-in can also be used. The web browser 52 of the client computing device 16 then attempts to render the requested web page 600 on a display of the client computing device 16.

[0060] While rendering the web page 600, and upon encountering the JavaScript code of the hotspot viewer plug-in 46, in step 402, the web browser 52 of the client computing device 16 executes the code to display the hotspot(s) of the image(s) included in the requested web page, such as the hotspot 604 of the image 602 of the web page 600 illustrated in FIG. 6. The hotspot 604 can be inserted into the image 602 of the web page 600 as described and illustrated above with respect to FIGS. 2-3, for example, although other methods of inserting the hotspot 604 can be used.

[0061] In step 404, the web browser 52 of the client computing device 16 executing the hotspot viewer plug-in 46 sends a JSONP request to the item preview web service 32 of the hotspot management computing apparatus 12 in response to a user interaction with one of the hotspots, such as hotspot 604. In this example, the JSONP request includes an identifier associated with the hotspot 604. The identifier can be the identifier stored as described with respect to the step 210, for example. Accordingly, the web page 600 obtained from the content server computing apparatus 14 in step 400 can include one of the stored identifiers as associated with each hotspot 604. Upon user interaction with the hotspot 604 of the image

602 of the web page 600, the JSONP request including the identifier is generated and sent to the item preview web service 32.

[0062] Optionally, the JSONP request further includes an attribute indicating a layout to be used, as described and illustrated in more detail below. The layout can be a default layout included in JavaScript code of the hotspot viewer plug-in 46, for example, although other methods of indicating a layout can also be used. Also optionally, the JSONP request can further include tracking information associated with the client computing device 16 executing the hotspot viewer plug-in 46.

[0063] The item preview web service 32 can be located at a URL including a different domain, host, port, or application layer protocol than the URL of the content server computing apparatus 14 from which the web page 600 was obtained in the step 400. The URL of the item preview web service 32 can be included in the hotspot viewer plug-in 44 such that user interaction with the hotspot 604 initiates the JSONP request to the item preview web service 46 located at the URL.

[0064] In step 406, the web browser 52 of the client computing device 16 executing the hotspot viewer plug-in 46 renders an HTML fragment received from and generated by the item preview web service 32, as described and illustrated in more detail below. The HTML fragment defines a preview panel, such as the exemplary preview panel 900 illustrated in FIG. 9. The content of the preview panel 900 can also be output to a display of the client computing device 16 in the form of a new window or overlay, for example, although any other form of display can also be used.

[0065] The HTML fragment is generated based on the identifier and defines at least a first, generally shorter, description associated with the hotspot and an input element, such as the input element 904 illustrated in FIG. 9. Accordingly, based on the identifier, the item preview web service 32 can obtain the content associated with the hotspot 604, such as the content associated with the selected search result 804(1). Upon obtaining the content, the item preview web service 32 can generate the HTML fragment according to the layout attribute value received in the JSONP request or a default layout, for example. In some

example, the input element 902 of the preview panel 900 is associated with a URL of a second web page including a second, generally longer, description associated with the hotspot 604, such as the remaining portion(s) of the content not displayed in the preview panel 900.

[0066] In step 408, the web browser 52 of the client computing device 16 executing the hotspot viewer plug-in 46 displays the second web page in response to a user interaction with the input element 904 of the preview panel 900. Optionally, the input element of the preview panel 900 is further associated with a viewer attribute. The viewer attribute can indicate the behavior to be used to generate the second web page (e.g. in a new panel or in a new window). Accordingly, in this example, the second web page is displayed on a display of the client computing device 16 according to the viewer attribute.

[0067] Referring to FIG. 5, an exemplary method of facilitating the display of a web page image hotspot is illustrated. In this example, in step 500, the hotspot management computing apparatus 12 implementing the item preview web service 32 receives a JSONP request from the hotspot viewer plug-in 46, such as the JSONP request sent in the step 404, for example. The JSONP request includes at least an identifier associated with a selected web page image hotspot, such as the hotspot 604 of the image 602 of the web page 600, for example. The item preview web service 32 of the hotspot management computing apparatus 12 can be located at a different domain, host, port, and/or application layer protocol than a URL of the content server computing apparatus 12 that provided the hotspot viewer plug-in 46 that originated the JSONP request.

[0068] In step 502, the hotspot management computing apparatus 12 implementing the item preview web service 32 generates an HTML fragment based on the identifier associated with the hotspot 604 sent along with the JSONP request. The HTML fragment defines a description associated with the hotspot 604, such the description associated with the content of the search result 804(1), as well as an input element 902, as described above. Additionally, in this example, the input element 902 is associated with a URL of a second web page including a second description associated with the selected hotspot 604 of the image 602 of the web page 600.

[0069] Optionally, the JSONP request further includes an indication of a layout to be used for presenting the preview content and the HTML fragment is generated according to the layout. Also optionally, the JSONP request includes tracking information associated with the client computing device 16 executing the hotspot viewer plug-in 46. The tracking information can be stored in the memory 24 of the hotspot management computing apparatus 12 for subsequent retrieval by an administrator, for example, although the tracking information can also be sent and/or stored elsewhere.

[0070] In step 504, the hotspot management computing apparatus 12 implementing the item preview web service 32 sends the HTML fragment to the requesting hotspot viewer plug-in 46 executed by the web browser 52 of the client computing device 16. The preview panel 900 shown in FIG. 9 is an example of a rendering by the web browser 52 of the client computing device 16 of the HTML fragment generated in step 504.

[0071] Accordingly, as illustrated and described herein this technology provides a number of advantages including improved methods, non-transitory computer readable medium, and devices for establishing and generating web page image hotspots. With this technology, inserting or editing an image hotspot of a web page can be facilitated by a JavaScript plug-in sent along with the web page. The JavaScript plug-in communicates with a JavaScript web service configured to locate content that can be associated with the hotspot. The JavaScript web service can be hosted by a hotspot management computing apparatus outside the domain of the content server apparatus hosting the web page.

[0072] Additionally, viewing image hotspots of a web page can be facilitated by a JavaScript plug-in sent along with the web page. The JavaScript plug-in communicates with a JavaScript web service configured to generate a preview panel to be displayed upon selection of one of the hotspots. The JavaScript web service can also be hosted by a hotspot management computing apparatus outside the domain of the content server apparatus hosting the web page.

[0073] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be

presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for facilitating interaction with a web page image hotspot, the method comprising:

receiving, by a hotspot management computing apparatus, a JavaScript Object Notation with Padding (JSONP) request from a client computing device, wherein the JSONP request comprises an identifier associated with a selected hotspot associated with an image of a first web page;

generating, by the hotspot management computing apparatus, a HyperText Markup Language (HTML) fragment based on the identifier, wherein the HTML fragment defines a first description associated with the hotspot and an input element associated with a Uniform Resource Locator (URL) of a second web page including a second description associated with the hotspot; and

sending, by the hotspot management computing apparatus, the HTML fragment to the requesting client computing device.

2. The method as set forth in claim 1, wherein the client computing device is a mobile computing device, the JSONP request further comprises an indication of a layout, and the HTML fragment is generated according to the layout.

3. The method as set forth in claim 1, wherein the JSONP request further comprises tracking information associated with the client computing device and the method further comprises storing, with the hotspot management computing apparatus, the tracking information as associated with the client computing device.

4. The method as set forth in claim 1, wherein one or more of the steps are performed by an item preview web service.

5. A computer readable medium having stored thereon instructions for

facilitating interaction with a web page image hotspot comprising machine executable code which when executed by a processor, causes the processor to perform steps comprising:

receiving a JavaScript Object Notation with Padding (JSONP) request from a client computing device, wherein the JSONP request comprises an identifier associated with a selected hotspot associated with an image of a first web page;

generating a HyperText Markup Language (HTML) fragment based on the identifier, wherein the HTML fragment defines a first description associated with the hotspot and an input element associated with a Uniform Resource Locator (URL) of a second web page including a second description associated with the hotspot; and

sending the HTML fragment to the requesting client computing device.

6. The medium as set forth in claim 5, wherein the client computing device is a mobile computing device, the JSONP request further comprises an indication of a layout, and the HTML fragment is generated according to the layout.

7. The medium as set forth in claim 5, wherein the JSONP request further comprises tracking information associated with the client computing device and the medium further has stored thereon instructions comprising machine executable code which when executed by the processor cases the processor to perform steps further comprising storing the tracking information as associated with the client computing device.

8. The medium as set forth in claim 5, wherein one or more of the steps are performed by an item preview web service.

9. A hotspot management computing apparatus, comprising:

a memory coupled to a processor which is configured to execute programmed instructions stored in the memory comprising:

receiving a JavaScript Object Notation with Padding (JSONP) request from a client computing device, wherein the JSONP request comprises an identifier

associated with a selected hotspot associated with an image of a first web page;

generating a HyperText Markup Language (HTML) fragment based on the identifier, wherein the HTML fragment defines a first description associated with the hotspot and an input element associated with a Uniform Resource Locator (URL) of a second web page including a second description associated with the hotspot; and

sending the HTML fragment to the requesting client computing device.

10. The apparatus as set forth in claim 9, wherein the client computing device is a mobile computing device, the JSONP request further comprises an indication of a layout, and the HTML fragment is generated according to the layout.

11. The apparatus as set forth in claim 9, wherein the JSONP request further comprises tracking information associated with the client computing device and the processor is further configured to execute programmed instructions stored in the memory further comprising storing the tracking information as associated with the client computing device.

12. The apparatus as set forth in claim 9, wherein one or more of the steps are performed by an item preview web service.

13. A method for interacting with a web page image hotspot, the method comprising:

displaying, with a client computing device, one or more hotspots of an image included on a first web page;

sending, with the client computing device, a JavaScript Object Notation with Padding (JSONP) request to a hotspot management computing apparatus in response to user interaction with one of the one or more hotspots, wherein the JSONP request comprises an identifier associated with the one hotspot;

rendering, with the client computing device, a HyperText Markup

Language (HTML) fragment received from the hotspot management computing apparatus in a preview panel, wherein the HTML fragment is based on the identifier and defines a first description associated with the one hotspot and an input element associated with a Uniform Resource Locator (URL) of a second web page including a second description associated with the one hotspot; and

displaying, with the client computing device, the second web page in response to user interaction with the input element of the preview panel.

14. The method as set forth in claim 13, wherein the client computing device is a mobile computing device, the JSONP request further comprises an indication of a layout, and the HTML fragment is rendered according to the layout.

15. The method as set forth in claim 13, wherein the JSONP request further comprises tracking information associated with the client computing device.

16. The method as set forth in claim 13, wherein the client computing device is a mobile computing device, the input element of the preview panel is further associated with a viewer attribute, and the second web page is displayed according to the viewer attribute.

17. The method as set forth in claim 13, wherein the JSONP request is sent to or the HTML fragment is received from an item preview web service of the hotspot management computing apparatus.

18. A non-transitory computer readable medium having stored thereon instructions for interacting with a web page image hotspot comprising machine executable code which when executed by a processor, causes the processor to perform steps comprising:

displaying one or more hotspots of an image included on a first web page;

sending a JavaScript Object Notation with Padding (JSONP) request to

a hotspot management computing apparatus in response to user interaction with one of the one or more hotspots, wherein the JSONP request comprises an identifier associated with the one hotspot;

rendering a HyperText Markup Language (HTML) fragment received from the hotspot management computing apparatus in a preview panel, wherein the HTML fragment is based on the identifier and defines a first description associated with the one hotspot and an input element associated with a Uniform Resource Locator (URL) of a second web page including a second description associated with the one hotspot; and

displaying the second web page in response to user interaction with the input element of the preview panel.

19. The medium as set forth in claim 18, wherein the client computing device is a mobile computing device, the JSONP request further comprises an indication of a layout, and the HTML fragment is rendered according to the layout.

20. The medium as set forth in claim 18, wherein the JSONP request further comprises tracking information associated with the client computing device.

21. The medium as set forth in claim 18, wherein the client computing device is a mobile computing device, the input element of the preview panel is further associated with a viewer attribute, and the second web page is displayed according to the viewer attribute.

22. The medium as set forth in claim 18, wherein the JSONP request is sent to or the HTML fragment is received from an item preview web service of the hotspot management computing apparatus.

23. A client computing device, comprising:

a memory coupled to a processor which is configured to execute programmed instructions stored in the memory comprising:

displaying one or more hotspots of an image included on a first web page;

sending a JavaScript Object Notation with Padding (JSONP) request to a hotspot management computing apparatus in response to user interaction with one of the one or more hotspots, wherein the JSONP request comprises an identifier associated with the one hotspot;

rendering a HyperText Markup Language (HTML) fragment received from the hotspot management computing apparatus in a preview panel, wherein the HTML fragment is based on the identifier and defines a first description associated with the one hotspot and an input element associated with a Uniform Resource Locator (URL) of a second web page including a second description associated with the one hotspot; and

displaying the second web page in response to user interaction with the input element of the preview panel.

24. The device as set forth in claim 23, wherein the client computing device is a mobile computing device, the JSONP request further comprises an indication of a layout, and the HTML fragment is rendered according to the layout.

25. The device as set forth in claim 23, wherein the JSONP request further comprises tracking information associated with the client computing device.

26. The device as set forth in claim 23, wherein the client computing device is a mobile computing device, the input element of the preview panel is further associated with a viewer attribute, and the second web page is displayed according to the viewer attribute.

27. The device as set forth in claim 23, wherein the JSONP request is sent to or the HTML fragment is received from an item preview web service of the hotspot management computing apparatus.

28. A method for generating a web page image hotspot, the method comprising:

receiving, with a hotspot management computing apparatus, a JavaScript Object Notation with Padding (JSONP) request from a client computing device, the JSONP request comprising search criteria;

identifying, by the hotspot management computing apparatus, one or more search results satisfying the search criteria;

generating, by the hotspot management computing apparatus, a HyperText Markup Language (HTML) fragment defining an input element associated with each of the one or more search results and an identifier, wherein each of the one or more search results includes content to be associated with a web page image hotspot when an associated one of the input elements is selected by a user of the client computing device; and

sending, by the hotspot management computing apparatus, the HTML fragment to the client computing device.

29. The method as set forth in claim 28, wherein the client computing device is a mobile computing device, the JSONP request further comprises an indication of a layout, and the HTML fragment is generated according to the layout.

30. The method as set forth in claim 28, wherein one or more of the steps are performed by an item search web service and the JSONP request is received from a hotspot editor plug-in retrieved by the client computing device from a content server computing apparatus located in a different domain than the item search web service.

31. A computer readable medium having stored thereon instructions for facilitating generation of a web page image hotspot comprising machine executable code which when executed by a processor, causes the processor to perform steps comprising:

receiving a JavaScript Object Notation with Padding (JSONP) request from a client computing device, the JSONP request comprising search criteria;

identifying one or more search results satisfying the search criteria;

generating a HyperText Markup Language (HTML) fragment defining an input element associated with each of the one or more search results and an identifier, wherein each of the one or more search results includes content to be associated with a web page image hotspot when an associated one of the input elements is selected by a user of the client computing device; and

sending the HTML fragment to the requesting client computing device.

32. The medium as set forth in claim 31, wherein the client computing device is a mobile computing device, the JSONP request further comprises an indication of a layout, and the HTML fragment is generated according to the layout.

33. The medium as set forth in claim 31, wherein one or more of the steps are performed by an item search web service and the JSONP request is received from a hotspot editor plug-in retrieved by the client computing device from a content server computing apparatus located in a different domain than the item search web service..

34. A hotspot management computing apparatus, comprising:

a memory coupled to one or more processors which are configured to execute programmed instructions stored in the memory comprising:

receiving a JavaScript Object Notation with Padding (JSONP) request from a client computing device, the JSONP request comprising search criteria;

identifying one or more search results satisfying the search criteria;

generating a HyperText Markup Language (HTML) fragment defining an input element associated with each of the one or more search results and an identifier, wherein each of the one or more search results includes content to be associated with a web page image hotspot when an associated one of the input elements is selected by a user of the client computing device; and

sending the HTML fragment to the requesting client computing

device.

35. The apparatus as set forth in claim 34, wherein the client computing device is a mobile computing device, the JSONP request further comprises an indication of a layout, and the HTML fragment is generated according to the layout.

36. The apparatus as set forth in claim 34, wherein one or more of the steps are performed by an item search web service and the JSONP request is received from a hotspot editor plug-in retrieved by the client computing device from a content server computing apparatus located in a different domain than the item search web service.

37. A method for generating a web page image hotspot, the method comprising:

generating, by a client computing device, a search panel in response to a user identifying a region of an image rendered on a web page as a hotspot, the search panel comprising one or more input fields;

sending, by the client computing device, based on interaction with the input element of the search panel, a JavaScript Object Notation with Padding (JSONP) request to a hotspot management computing apparatus, wherein the JSONP request comprises search criteria entered in the one or more input fields;

receiving, by the client computing device, a HyperText Markup Language (HTML) fragment from the hotspot management computing apparatus in response to the JSONP request;

rendering, by the client computing device, the HTML fragment in a results panel, the HTML fragment defining an input element associated with each of one or more search results, wherein each input element is associated with an identifier of one of the search results; and

sending, by the client computing device, the identifier associated with a selected one of the input elements to the hotspot management computing apparatus.

38. The method as set forth in claim 37, wherein the client computing device is a mobile computing device, the JSONP request further comprises an indication of a layout, and the HTML fragment is rendered according to the layout.

39. The method as set forth in claim 37, wherein the JSONP request is sent to an item search web service of the hotspot management computing apparatus by a hotspot editor plug-in retrieved from a content server computing apparatus located in a different domain than the item search web service.

40. A non-transitory computer readable medium having stored thereon instructions for generating a web page image hotspot comprising machine executable code which when executed by a processor, causes the processor to perform steps comprising:

generating a search panel in response to a user identifying a region of an image rendered on a web page as a hotspot, the search panel comprising one or more input fields;

sending based on interaction with the input element of the search panel a JavaScript Object Notation with Padding (JSONP) request to a hotspot management computing apparatus, wherein the JSONP request comprises search criteria entered in the one or more input fields;

receiving, by the client computing device, a HyperText Markup Language (HTML) fragment from the hotspot management computing apparatus in response to the JSONP request;

rendering the HTML fragment in a results panel, the HTML fragment defining an input element associated with each of one or more search results, wherein each input element is associated with an identifier of one of the search results; and

sending the identifier associated with a selected one of the input elements to the hotspot management computing apparatus.

41. The medium as set forth in claim 40, wherein the client computing device is a mobile computing device, the JSONP request further comprises an indication of a

layout, and the HTML fragment is rendered according to the layout.

42. The medium as set forth in claim 40, wherein the JSONP request is sent to an item search web service of the hotspot management computing apparatus by a hotspot editor plug-in retrieved from a content server computing apparatus located in a different domain than the item search web service.

43. A client computing device, comprising:

a memory coupled to a processor which is configured to execute programmed instructions stored in the memory comprising:

generating a search panel in response to a user identifying a region of an image rendered on a web page as a hotspot, the search panel comprising one or more input fields;

sending based on interaction with the input element of the search panel a JavaScript Object Notation with Padding (JSONP) request to a hotspot management computing apparatus, wherein the JSONP request comprises search criteria entered in the one or more input fields;

receiving, by the client computing device, a HyperText Markup Language (HTML) fragment from the hotspot management computing apparatus in response to the JSONP request;

rendering the HTML fragment in a results panel, the HTML fragment defining an input element associated with each of one or more search results, wherein each input element is associated with an identifier of one of the search results; and

sending the identifier associated with a selected one of the input elements to the hotspot management computing apparatus.

44. The device as set forth in claim 43, wherein the client computing device is a mobile computing device, the JSONP request further comprises an indication of a layout, and the HTML fragment is rendered according to the layout.

45. The device as set forth in claim 43, wherein the JSONP request is sent to an item search web service of the hotspot management computing apparatus by a hotspot editor plug-in retrieved from a content server computing apparatus located in a different domain than the item search web service.

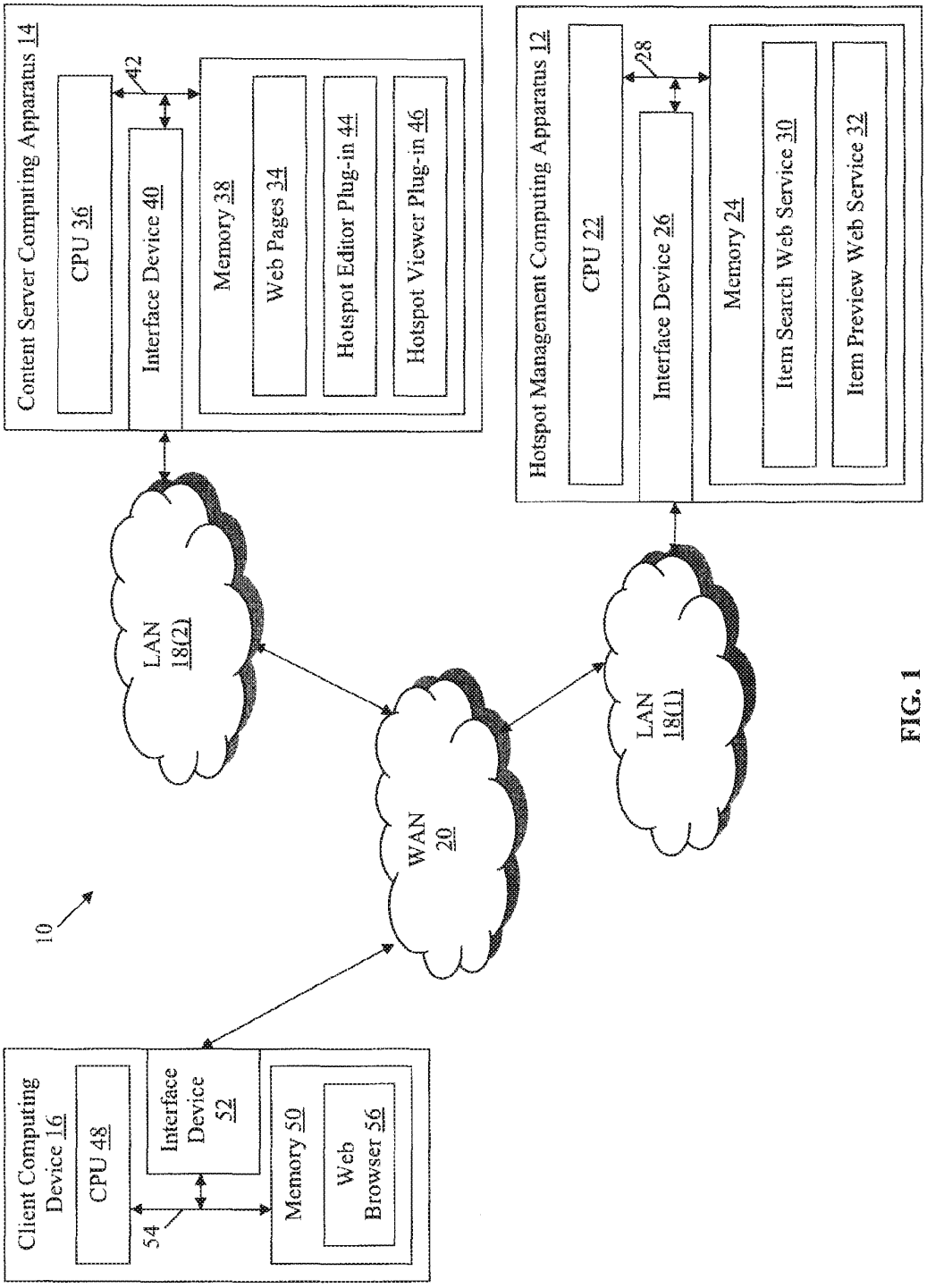

Content Server Computing Apparatus 14
CPU 36
42
Interface Device 40
Memory 38
Web Pages 34
Hotspot Editor Plug-in 44
Hotspot Viewer Plug-in 46
Hotspot Management Computing Apparatus 12
CPU 22
28
Interface Device 26
Memory 24
Item Search Web Service 30
Item Preview Web Service 32
LAN 18(2)
LAN 18(1)
WAN 20
10
Client Computing Device 16
CPU 48
54
Interface Device 52
Memory 50
Web Browser 56

FIG. 1

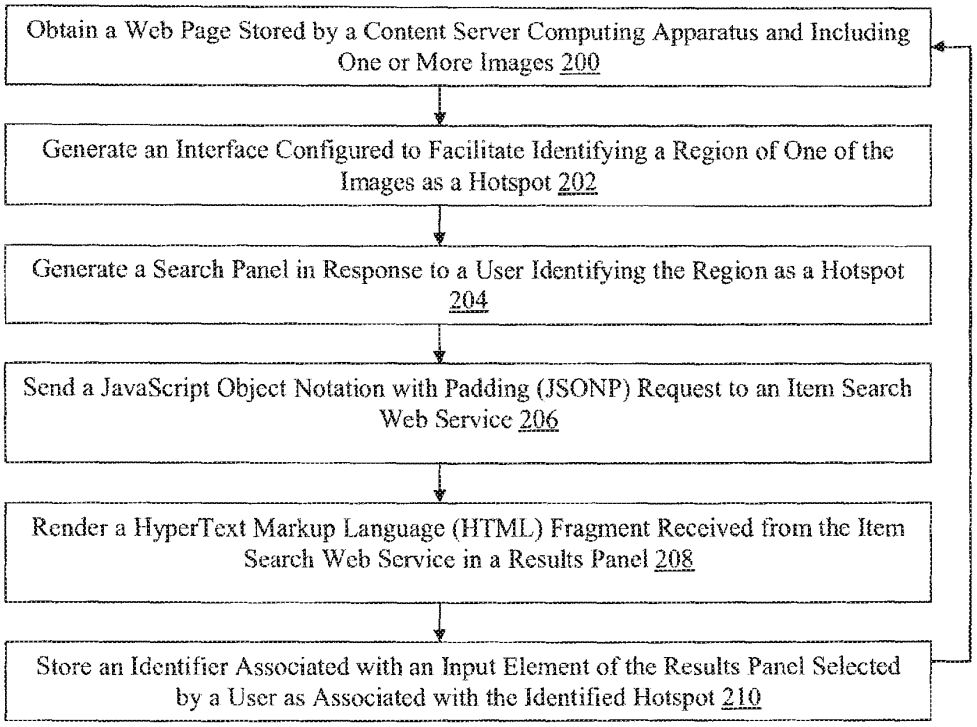

Obtain a Web Page Stored by a Content Server Computing Apparatus and Including One or More Images 200
Generate an Interface Configured to Facilitate Identifying a Region of One of the Images as a Hotspot 202
Generate a Search Panel in Response to a User Identifying the Region as a Hotspot 204
Send a JavaScript Object Notation with Padding (JSONP) Request to an Item Search Web Service 206
Render a HyperText Markup Language (HTML) Fragment Received from the Item Search Web Service in a Results Panel 208
Store an Identifier Associated with an Input Element of the Results Panel Selected by a User as Associated with the Identified Hotspot 210

FIG. 2

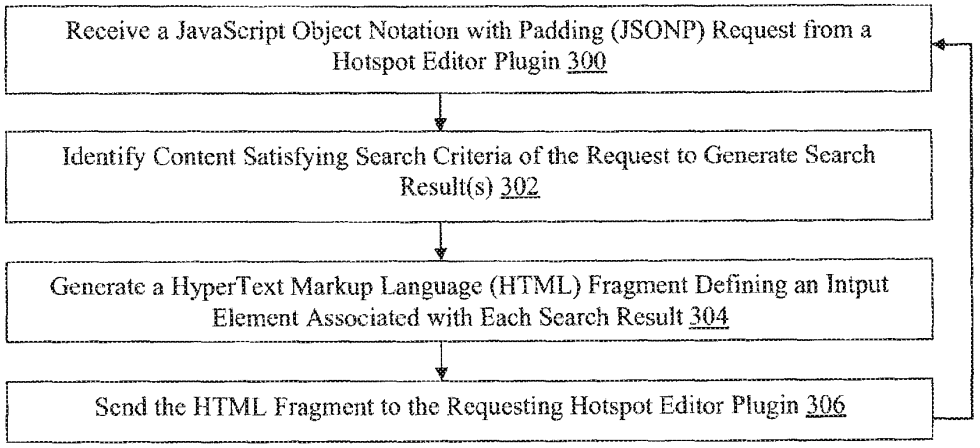

Receive a JavaScript Object Notation with Padding (JSONP) Request from a Hotspot Editor Plugin 300
Identify Content Satisfying Search Criteria of the Request to Generate Search Result(s) 302
Generate a HyperText Markup Language (HTML) Fragment Defining an Intput Element Associated with Each Search Result 304
Send the HTML Fragment to the Requesting Hotspot Editor Plugin 306

FIG. 3

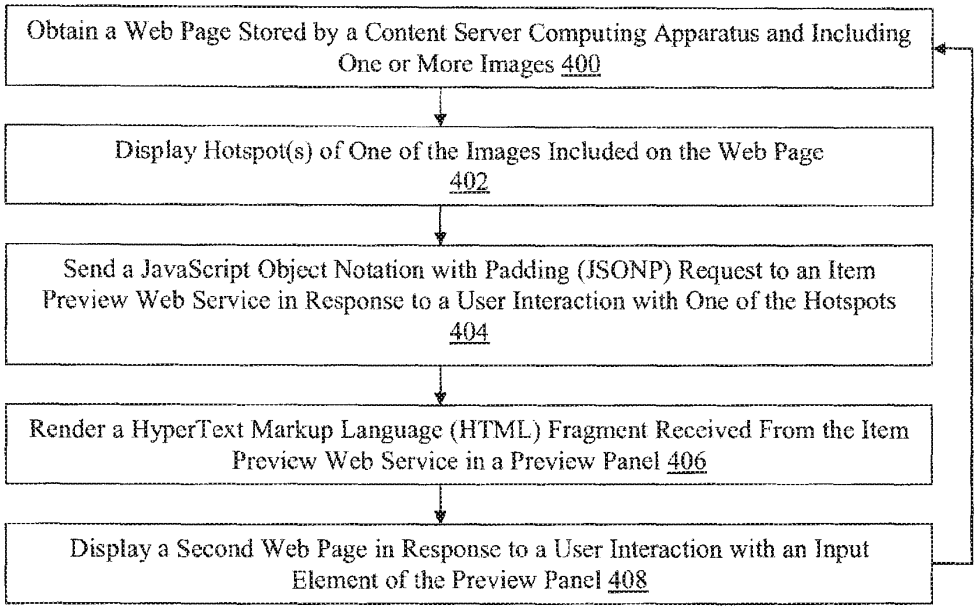

Obtain a Web Page Stored by a Content Server Computing Apparatus and Including One or More Images 400
Display Hotspot(s) of One of the Images Included on the Web Page 402
Send a JavaScript Object Notation with Padding (JSONP) Request to an Item Preview Web Service in Response to a User Interaction with One of the Hotspots 404
Render a HyperText Markup Language (HTML) Fragment Received From the Item Preview Web Service in a Preview Panel 406
Display a Second Web Page in Response to a User Interaction with an Input Element of the Preview Panel 408

FIG. 4

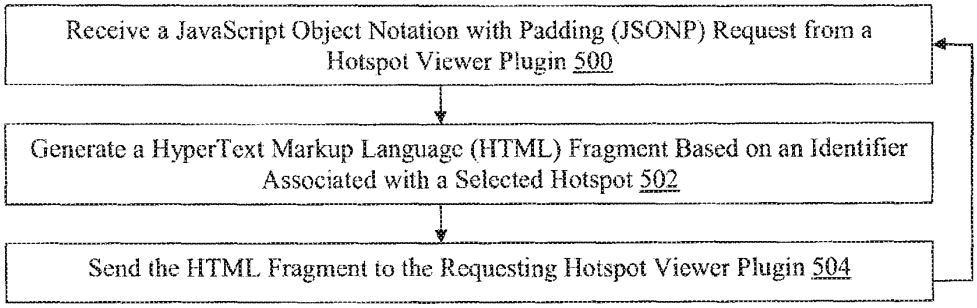

Receive a JavaScript Object Notation with Padding (JSONP) Request from a Hotspot Viewer Plugin 500
Generate a HyperText Markup Language (HTML) Fragment Based on an Identifier Associated with a Selected Hotspot 502
Send the HTML Fragment to the Requesting Hotspot Viewer Plugin 504

FIG. 5

FIG. 6

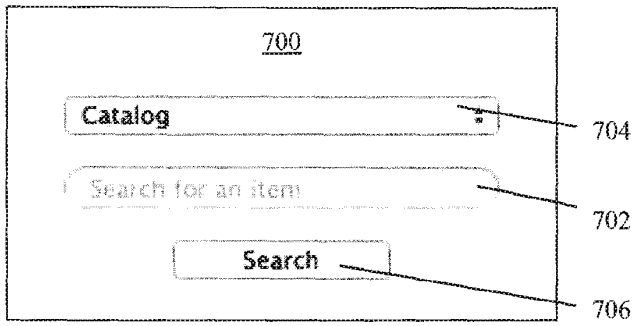

700
Catalog
Search for an item
Search
704
702
706

FIG. 7


800
804(1)
804(2)
Double Handle Bowler Bag
£32.50
£45.00
Strapless Skater Dress in Lace
Select
Select
802(1)
802(2)

FIG. 8

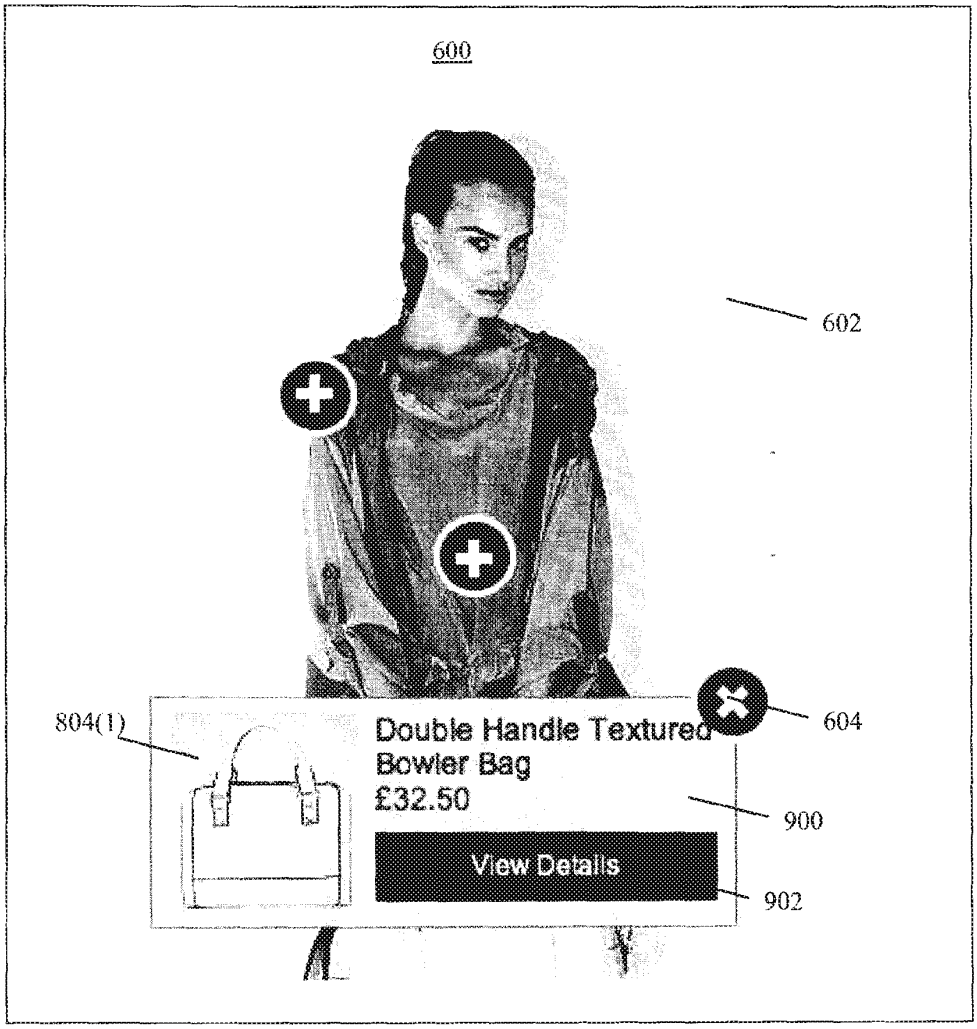
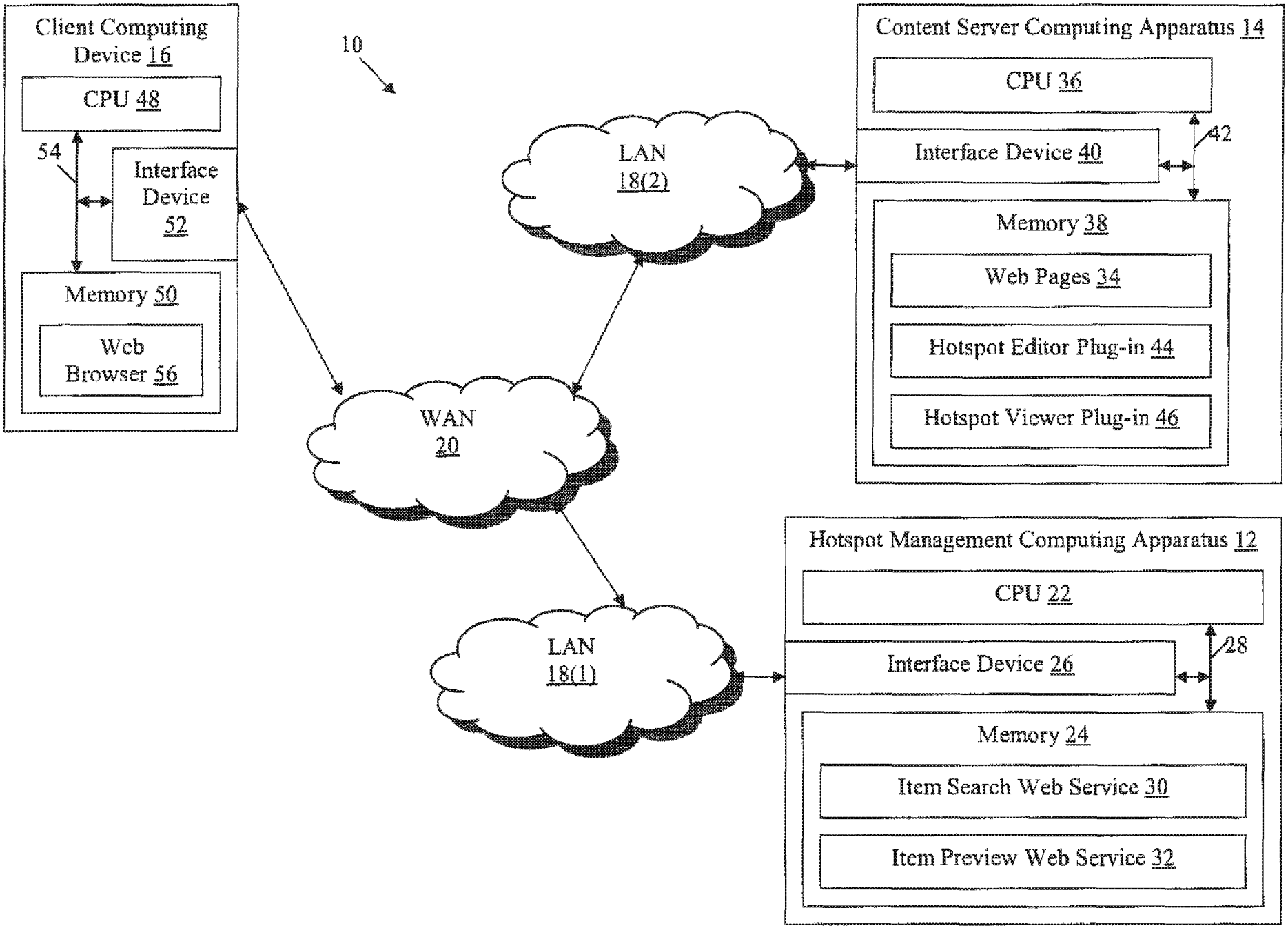

600
602
804(1)
Double Handle Textured
Bowler Bag
£32.50
604
900
View Details
902

FIG. 9



Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2834466 A1 2014/05/29

(21) 2 834 466

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(22) Date de dépôt/Filing Date: 2013/11/27

(41) Mise à la disp. pub./Open to Public Insp.: 2014/05/29

(30) Priorité/Priority: 2012/11/29 (US61/731,205)

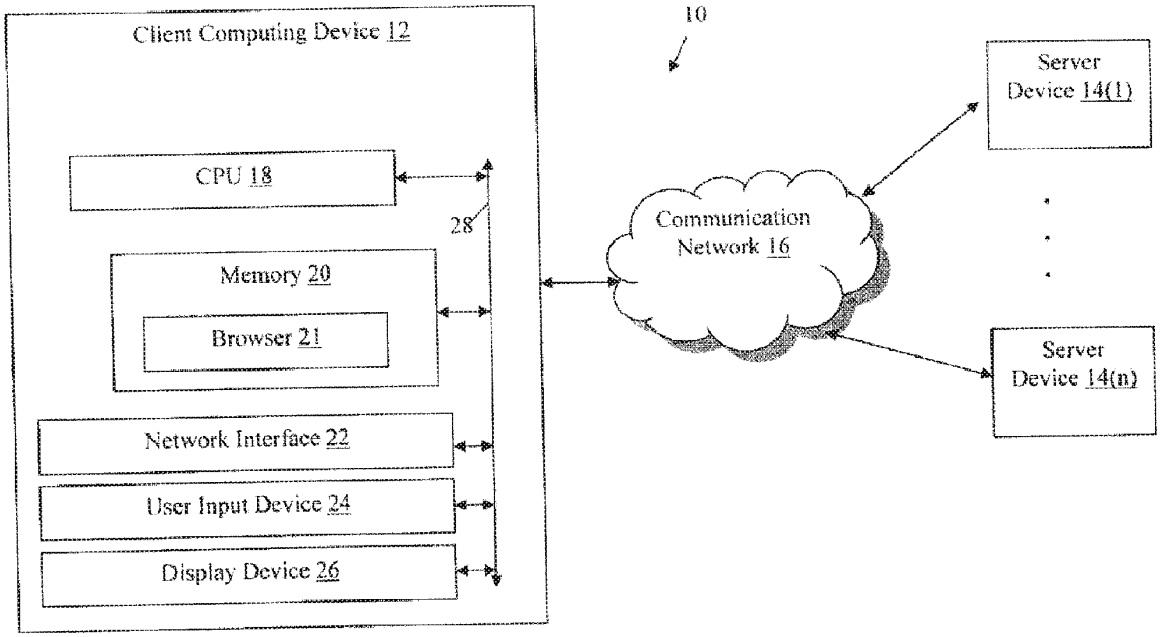
(51) Cl.Int./Int.Cl. *G06F 17/30* (2006.01), *H04L 12/16* (2006.01), *G06Q 10/02* (2012.01)

(71) Demandeur/Applicant:
USABLENET INC., US

(72) Inventeur/Inventor:
SCODA, ENRICO, US

(74) Agent: PARLEE MCLAWS LLP

(54) Titre : METHODES DE LIVRAISON DE SUGGESTION DE RECHERCHE WEB ET DISPOSITIFS ASSOCIES
(54) Title: METHODS FOR PROVIDING WEB SEARCH SUGGESTIONS AND DEVICES THEREOF



(57) Abrégé/Abstract:
A method, non-transitory computer readable medium, and programmed device that use local storage in a browser to provide one or more web search suggestions includes receiving at least a partial entry in field of a web page. Any responsive web search suggestion entries to the received at least a partial entry are provided in the field of the web page from a web search suggestion file stored in a web browser of the client computing device.

ABSTRACT

A method, non-transitory computer readable medium, and programmed device that use local storage in a browser to provide one or more web search suggestions includes receiving at least a partial entry in field of a web page. Any responsive web search suggestion entries to the received at least a partial entry are provided in the field of the web page from a web search suggestion file stored in a web browser of the client computing device.

METHODS FOR PROVDING WEB SEARCH SUGGESTIONS AND DEVICES THEREOF

CROSS REFERENCE

[0001] This application claims the benefit of U.S. Provisional Patent Application Serial No. 61/731,205 filed November 29, 2012, which is hereby incorporated by reference in its entirety.

FIELD

[0002] This technology generally relates to methods and devices for assisting web searching and, more particularly, to methods for using local storage in a browser at a client computing device for providing one or more web search suggestions and devices thereof.

BACKGROUND

[0003] Web sites improve the user experience around "search item" functionalities by predicting a word or a phrase when a user at a client computing device starts typing. By way of example, when the user at the client computing device starts typing the beginning of the word, such as "New" for a city, an airport, or a rail station, then the Web content provider could provide suggestions, such as "New York" and "New Heaven". In another example, when the user at the client computing device starts typing "soc", then the Web content provider could provide suggestions, such as "socks 15" and "soccer 3" where the numbers indicate the number of items in the catalog including the suggested words.

[0004] This search suggestion functionality is a common user interface feature that is usually implemented with Ajax technology at the client computing device.

When a user at the client computing device types in a text field, the JavaScript engine in the client computing devices sends a request to a Web content provider. This Web content provider returns a data set, such as JSON data set, that the JavaScript at the client computing device will use to display results as a list of suggestions.

[0005] When the internet connection for the client computing device is fast, i.e. low latency, this search suggestion functionality works pretty well. However when the internet connection for the client computing device is slow or intermittent, such as when browsing the Web from a mobile computing device, then the performance of this search suggestion service often deteriorates.

SUMMARY

[0006] A method using local storage in a browser to provide one or more web search suggestions includes receiving by a client computing device at least a partial entry in field of a web page. Any responsive web search suggestion entries to the received at least a partial entry are provided by the client computing device in the field of the web page from a web search suggestion file stored in a web browser of the client computing device.

[0007] A non-transitory computer readable medium having stored thereon instructions for providing one or more web search suggestions comprising machine executable code which when executed by at least one processor, causes the processor to perform steps comprising receiving at least a partial entry in field of a web page. Any responsive web search suggestion entries to the received at least a partial entry are provided in the field of the web page from a web search suggestion file stored in a web browser of the client computing device.

[0008] A computing device includes a memory coupled to one or more processors which are configured to execute programmed instructions stored in the memory including receiving at least a partial entry in field of a web page. Any responsive web search suggestion entries to the received at least a partial entry are provided in the field of the web page from a web search suggestion file stored in a web browser of a client computing device.

[0009] This technology provides a number of advantages including providing methods, non-transitory computer readable medium, and devices that more quickly and effectively provide web search suggestions. In particular, with for example HTML5, this technology can utilize local storage within a browser to store a web search suggestion file for use in providing web search suggestions. As a result, in environments where the internet connection is slower, i.e. has higher latency, this technology is able to more quickly provide web search suggestions.

BRIEF DESCRIPTION OF THE DRAWINGS

[00010] FIG. 1 is a block diagram of an environment with a client computing device which uses local storage in a browser to facilitate providing web search suggestions;

[00011] FIG. 2 is a flow chart of an example of a method for using local storage in a browser to provide one or more web search suggestions;

[00012] FIG. 3A is a screen shot of an example of a web page for ACME airlines with two entry fields for a flight reservation; and

[00013] FIG. 3B is a screen shot of an example of web search suggestions provided from the web search suggestion file in local storage in the browser of the client computing device.

DETAILED DESCRIPTION

[00014] An exemplary environment 10 with a client computing device 12 which uses local storage in a browser to facilitate providing web search suggestions is illustrated in FIG. 1. In this example, the client computing device 12 is coupled to one or more server devices 14(1)-14(n) by a communication network 16, although other types and numbers of systems, devices, components, and/or elements in other topologies could be used. This technology provides a number of advantages including providing methods, non-transitory computer readable medium, and devices that more quickly and effectively provide web search suggestions.

[00015] Referring more specifically to FIG. 1, the client computing device 12 is a mobile phone, although other types and numbers of client computing devices may be used, such as computer tablets, laptop computers and desktop computers by way of example only. In this example, the client computing device 12 includes at least one processor or central processing unit (CPU) 18, a memory 20 with a browser 21, a network interface 22, a user input device 24, and a display device 26 which are coupled together by a bus 28 or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can also be used. Generally, the client computing device 12 may for example request and receive web pages and other web content from one or more of the server devices 14(1)-14(n) via the communication network 16 according to the HTTP-based protocol, for example,

although the client computing device 12 can have and provide other types and numbers of functions and other operations. The processor 18 in the client computing device 12 may execute a program of stored instructions one or more aspects of the present invention, as described and illustrated by way of the embodiments herein, although the processor 18 could execute other numbers and types of programmed instructions.

[00016] The memory 20 in the client computing device 12 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a RAM or a ROM in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other non-transitory computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor 18, can be used for the memory 20 in the client computing device 12. In this example, the memory includes the browser 21 which may comprise any application configured to for example retrieve, present and/or traverse information resources and other content on the world wide web. With HTML 5, data, such as a web search suggestion file, can be locally stored within the browser 21, although other manners of obtaining local storage with the browser could be used.

[00017] The network interface 22 in the client computing device 12 is used to operatively couple and communicate between the client computing device 12 and the server devices 14(1)-14(n) via the communication network 16, although other types and numbers of networks with other types and numbers of connections and configurations can also be used.

[00018] The user input device 24 in the client computing device 12 can be used to input selections, such as a request for a particular web page or to enter data into a field of a web page, although the user input device could be used to input other types of requests and data and interact with other elements. The user input device in the client computing device 12 can include a keypad, touch screen, and/or vocal input processing system, although other types and numbers of user input devices can also be used.

[00019] The display device 26 in the client computing device 12 can be used to show data and other information to the user, such as a requested web page by way of example only. The display device 26 in in the client computing device 12 can be an LCD, LED, or OLED display, for example, although other types and numbers of displays could be used depending on the particular type of client computing device 12.

[00020] The communication network 16 can include one or more networks, such as one or more wide area networks (WANs), for example the Internet, and/or one or more local area networks (LANs). By way of example only, the communication network 16 can use TCP/IP over Ethernet and industry-standard protocols, including Hypertext transfer protocol (HTTP), secure HTTP (HTTPS), wireless application protocol (WAP), and/or SOAP, although other types and numbers of communication networks having their own communications protocols, can also be used.

[00021] The server devices 14(1)-14(n) may each one or more host applications, web services, and/or other data and content which may be requested and retrieved by the client computing device 12 via the communication network 16, although the server

devices could each provide a wide variety of other types of functions and other operations.

[00022] In this example, each of the server devices 14(1)-14(n) includes at least one processor or a CPU, a memory, and a network interface, which are coupled together by a bus or other link, although one or more of server devices 14(1)-14(n) can include other numbers and types of systems, devices, components, or other elements in other configurations. The processor in each of the server devices 14(1)-14(n) can execute a program of instructions stored in the memory of each of the server devices 14(1)-14(n) for one or more aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions.

[00023] The memory in each of the server devices 14(1)-14(n) stores these programmed instructions for one or more aspects of the present invention, as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a RAM or a ROM in the system or a floppy disk, hard disk, CD ROM, or other non-transitory computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor can be used for the memory in each of the server devices 14(1)-14(n).

[00024] The network interface in each of the server devices 14(1)-14(n) is used to operatively couple and communicate between each of the server devices 14(1)-14(n) and the client computing device 12 via the communication network 16, although other types

and numbers of communication networks with other types and numbers of connections and configurations can be used.

[00025] Although embodiments of the client computing device 12 and server devices 14(1)-14(n) are described and illustrated herein, each of the client computing device 12 and the server devices 14(1)-14(n) can be implemented on any suitable computer apparatus or computing device. It is to be understood that the apparatuses and devices of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s). Furthermore, each of the devices of the embodiments may be conveniently implemented using one or more general purpose computers, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[00026] In addition, two or more computing apparatuses or devices can be substituted for any one of the devices in any embodiment described herein. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices of the embodiments. The embodiments may also be implemented on computer apparatuses or devices that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public

Switched Telephone Networks (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[00027] The examples may also be embodied as a non-transitory computer readable medium having instructions stored thereon for one or more aspects of the present technology as described and illustrated by way of the examples herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the examples, as described and illustrated herein.

[00028] An exemplary method for using local storage in a client computing device 12 for one or more web search suggestions will now be described with reference to FIGS. 1-3B. In step 100, the client computing device 12 requests a web page of a web site from one of the server devices 14(1)-14(n), although other types of content could be request. The one of the web content server devices 14(1)-14(n) which hosts the web site responds with the requested web page as well as a current version of web search suggestion file for the web site which is received by the client computing device 12 in step 102.

[00029] In step 104, the requesting client computing device 12 determines whether the browser has local storage for the received version of the web search suggestion file. In this example, with HTML 5 the browser 21 in the requesting computing device 12 can provide local storage so the Yes branch is taken to step 106. In step 106, the received version of the web search suggestion file is stored locally within the browser 21 of the client computing device 12 and in this example without the use of a cookie, although other manners of storing locally within the browser 21 can be used. More specifically and

by way of example only, once the web search suggestion file is downloaded to the client computing device 12, a set of records can be inserted in to one or more tables in a database in browser 21 to assist with future queries, although other manners for organizing and locally storing the web search suggestion file can be used. If in step 104, the requesting computing device 12 cannot provide local storage, then the No branch is taken to step 120 where the user of the client computing device 12 can continue with any viewing or other interaction with the received web page.

[00030] In step 108, the requesting client computing device 12 may optionally determine whether to update any stored version of a web search suggestion file. By way of example only, the requesting client computing device 12 may adhere to a predefined and stored synchronization protocol for updating any stored version of a web search suggestion file, although other manners for maintaining updates of any locally stored web search suggestion file at the requesting one of the client computing devices can be used.

[00031] If in step 108 the requesting client computing device 12 determines to update any stored version of a web search suggestion file, then the Yes branch is taken to step 110. In step 110, the client computing device 12 will request and receive from the one of the server devices 14(1)-14(n) the updated web search suggestion file which is stored locally within the browser 21 in the client computing device 12. If in step 108 the requesting client computing device 12 determines not to update any stored version of a web search suggestion file, then the No branch is taken to step 112.

[00032] In addition to any optional periodic synchronization, anytime the client computing device 12 submits another request for a web page of the web site, the

client computing device 12 may optionally perform a version check to compare the current locally stored version of the web search suggestion file for the web site against the version of the web search suggestion file for the web site at the one of the web content server devices 14(1)-14(n) which hosts the web site. If the versions do not match, then the new web search suggestion file may be requested and downloaded from the one of the web content server devices 14(1)-14(n) to the client computing device 12 updating the locally stored database instance in browser 21.

[00033] In step 112, text may be typed into a text field of a received web page using user input device 24, although other manners for making an entry in other locations of a web page could be used. By way of example only, the text of "New" is typed into a "From" text field 202(1) with a currently empty "To" text field 202(2) of a received web page 200 for "ACME Airlines" for making flight reservations as illustrated in FIG. 3A.

[00034] In step 114, when text is entered into a field of a received web page on display device 26 of the client computing device 12, such as the text of "New" in the "From" text field 202(1), then in this example a JavaScript code executes in the client computing device 12 to determine if a database instance of the web search suggestion file is in the local storage within browser 21, although other types of processes could be used. If in step 114, the client computing device 12 determines a database instance of the web search suggestion file is not stored in the local storage in browser 21, then the No branch is taken to step 120 as described earlier. If in step 114, the client computing device 12 determines a database instance of the web search suggestion file is stored in the local storage in browser 21, then the Yes branch is taken to step 116.

[00035] In step 116, in this example the client computing device 12 executes an SQL query to retrieve all the rows from the stored a database instance of the web search suggestion file within browser 21 containing the text that was entered, although other manners for querying the locally stored web search suggestion file could be used. More specifically, in this particular illustrative example two types of queries could be performed by the client computing device 12, although other types and numbers of queries could be utilized. One type of query retrieves all rows that include all words entered by the user at the client computing device 12 in the version of the web search suggestion file stored by utilizing HTML 5 functionalities. This is a simple query and can be used when the number of results is not high. For example, given the entry of the word "New" entered in the text field 202(1) in FIG.3A, the client computing device 12 would retrieve or suggest all text containing this string, which in this particular example might suggest for the text "New" in the text field 202(1) in the web page "New York", "New Haven", or "New Plymouth" as illustrated in FIG. 3B by way of example only

[00036] Another type of optional query retrieves all rows that include the string entered by the user at the client computing device 12. Accordingly, this type of query retrieves records that include the entered string entered in the version of the web search suggestion file stored at the client computing device 12 by utilizing HTML 5 functionalities. For example, given a string "soc" entered in a text field which is searched, the client computing device 12 might retrieve, group and suggest items, such as "soccer" (2) or "socks" (8), where the number in parenthesis represents the number of records including the previous string. This technique is more CPU intensive and time consuming,

however it provides a better user experience at the client computing device 12 especially when applied to large catalogs or other data.

[00037] In step 118, the client computing device 12 provides the suggestions identified from the locally stored web search suggestion file for the text field 202(1) as illustrated in FIG. 3B in this example, although other manners for providing the suggestions could be used. In step 120, the client computing device 12 can continue to view and interact with the web page on the display device 26.

[00038] Accordingly, as illustrated and described with the examples herein this technology provides methods, non-transitory computer readable medium, and devices that more quickly and effectively provide web search suggestions. In these examples this technology uses the local storage capabilities provided by HTML5 at the client computing device. With this technology, the process of predicting words or phrases can be applied directly and efficiently in the browser of the client computing device.

[00039] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for providing one or more web search suggestions, the method comprising:

receiving, by a client computing device, at least a partial entry in field of a web page;

providing, by the client computing device, in the field of the web page any responsive web search suggestion entries to the received at least a partial entry from a web search suggestion file stored in a web browser of the client computing device.

2. The method of claim 1 further comprising determining, by the client computing device, whether the web search suggestion file is stored in the web browser of the client computing device, wherein the providing in the field of the web page any responsive web search suggestion entries occurs when the determining indicates the web search suggestion file is stored in the web browser of the client computing device.

3. The method as set forth in claim 2 further comprising:

determining, by the client computing device, whether the web search suggestion file stored in the web browser has an updated version; and

obtaining and storing, by the client computing device, the updated version in the web browser at the client computing device when the determining indicates the update version is available.

4. The method as set forth in claim 1 further comprising utilizing, by the client computing device, HTML5 to store the web search suggestion file in the web browser of the client computing device.

5. The method as set forth in claim 1 wherein the providing further comprises:

obtaining, by the client computing device, any responsive web search suggestion entries to the at least a partial entry from the web search suggestion file;

grouping, by the client computing device, any of the obtained responsive web search suggestion entries based on at least one common characteristic; and

providing, by the client computing device, the grouped responsive web search suggestion entries with a number in each group corresponding to a number of the responsive web search suggestion entries.

6. A non-transitory computer readable medium having stored thereon instructions for providing one or more web search suggestions comprising machine executable code which when executed by at least one processor, causes the processor to perform steps comprising:

receiving at least a partial entry in field of a web page; and providing in the field of the web page any responsive web search suggestion entries to the received at least a partial entry from a web search suggestion file stored in a web browser of the client computing device.

7. The medium of claim 6 further comprising determining whether the web search suggestion file is stored in the web browser of the client computing device, wherein the providing in the field of the web page any responsive web search suggestion entries occurs when the determining indicates the web search suggestion file is stored in the web browser of the client computing device.

8. The medium as set forth in claim 7 further comprising: determining whether the web search suggestion file stored in the web browser has an updated version; and obtaining and storing the updated version in the web browser at the client computing device when the determining indicates the update version is available.

9. The medium as set forth in claim 6 further comprising utilizing HTML5 to store the web search suggestion file in the web browser of the client computing device.

10. The medium as set forth in claim 6 wherein the providing further comprises:

obtaining any responsive web search suggestion entries to the at least a partial entry from the web search suggestion file;

grouping any of the obtained responsive web search suggestion entries based on at least one common characteristic; and

providing the grouped responsive web search suggestion entries with a number in each group corresponding to a number of the responsive web search suggestion entries.

11. A computing device comprising:

one or more processors;

a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory comprising:

receiving at least a partial entry in field of a web page; and

providing in the field of the web page any responsive web search suggestion entries to the received at least a partial entry from a web search suggestion file stored in a web browser of the client computing device.

12. The device as set forth in claim 11 wherein the memory coupled to the one or more processors is further configured to execute programmed instructions stored in the memory further comprising determining whether the web search suggestion file is stored in the web browser of the client computing device, wherein the providing in the field of the web page any responsive web search suggestion entries occurs when the determining indicates the web search suggestion file is stored in the web browser of the client computing device.

13. The device as set forth in claim 12 wherein the memory coupled to the one or more processors is further configured to execute programmed instructions stored in the memory further comprising:

determining whether the web search suggestion file stored in the web browser has an updated version; and

obtaining and storing the updated version in the web browser at the client computing device when the determining indicates the update version is available.

14. The device as set forth in claim 11 wherein the memory coupled to the one or more processors is further configured to execute programmed instructions stored in the memory further comprising utilizing HTML5 to store the web search suggestion file in the web browser of the client computing device.

15. The device as set forth in claim 11 wherein the memory coupled to the one or more processors is further configured to execute programmed instructions stored in the memory for the providing further comprises:

obtaining any responsive web search suggestion entries to the at least a partial entry from the web search suggestion file;

grouping any of the obtained responsive web search suggestion entries based on at least one common characteristic; and

providing the grouped responsive web search suggestion entries with a number in each group corresponding to a number of the responsive web search suggestion entries.

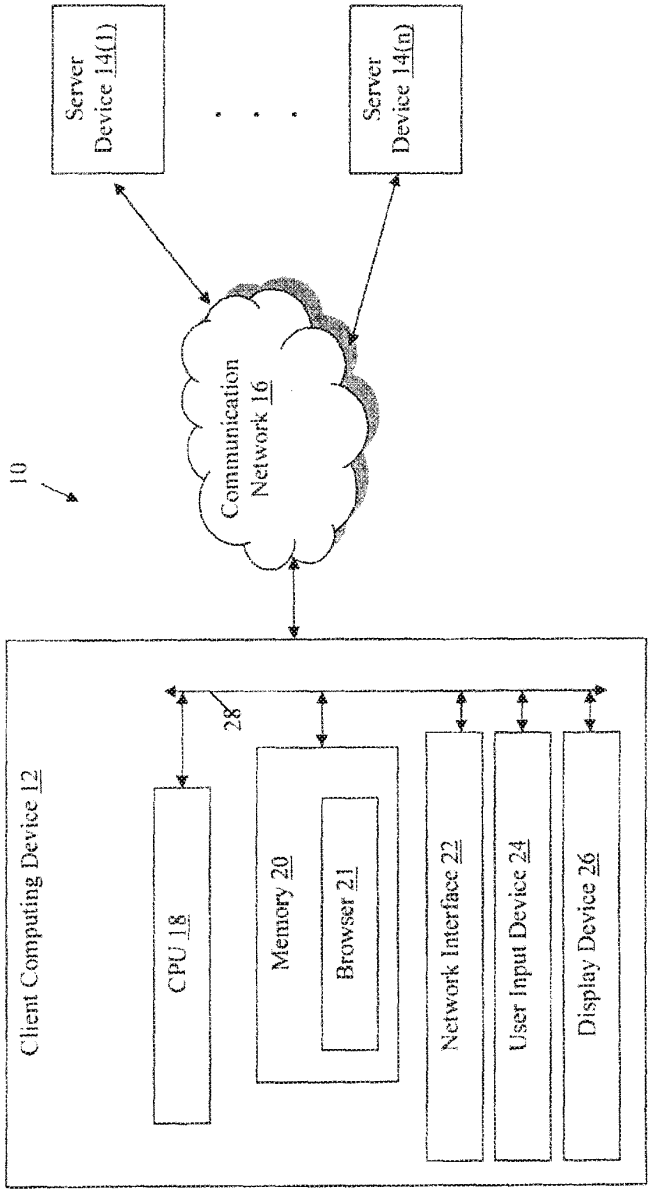


FIG. 1

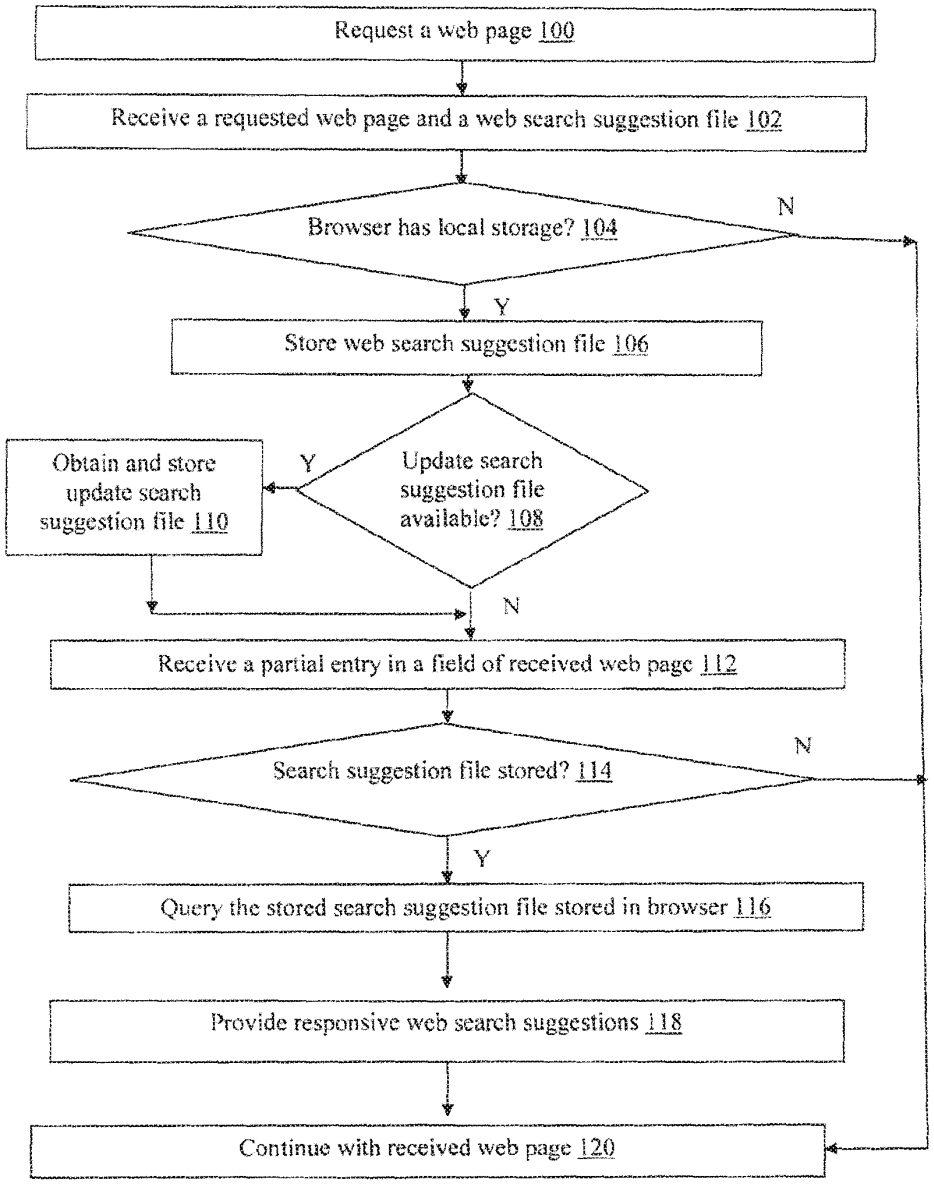


FIG. 2

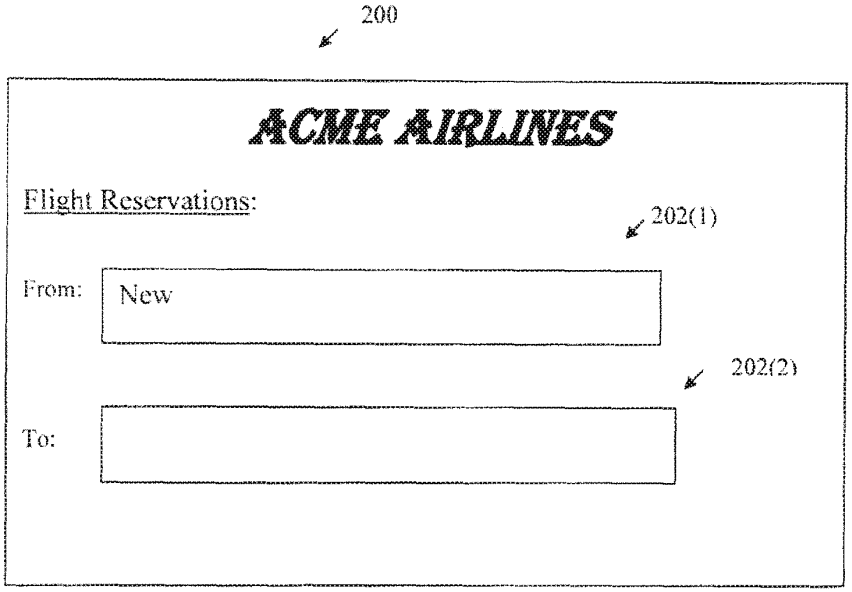

200
ACME AIRLINES
Flight Reservations:
202(1)
From:
New
202(2)
To:

FIG. 3A

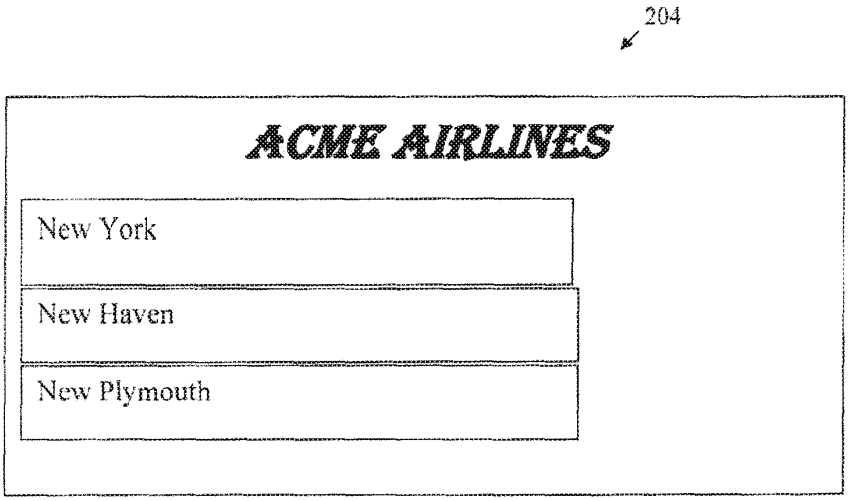

204
ACME AIRLINES
New York
New Haven
New Plymouth

FIG. 3B

Server Device 14(1)
Server Device 14(n)
Communication Network 16
10
Client Computing Device 12
CPU 18
28
Memory 20
Browser 21
Network Interface 22
User Input Device 24
Display Device 26

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2839006 C 2017/08/01

(11)(21) 2 839 006

(12) BREVET CANADIEN
CANADIAN PATENT

(13) C

(86) Date de dépôt PCT/PCT Filing Date: 2012/06/25
(87) Date publication PCT/PCT Publication Date: 2012/12/27
(45) Date de délivrance/Issue Date: 2017/08/01
(85) Entrée phase nationale/National Entry: 2013/12/10
(86) N° demande PCT/PCT Application No.: US 2012/044012
(87) N° publication PCT/PCT Publication No.: 2012/178167
(30) Priorité/Priority: 2011/06/24 (US13/135,112)

(51) Cl.Int./Int.Cl. *G06F 17/00* (2006.01), *G06F 15/16* (2006.01), *G06F 17/20* (2006.01)

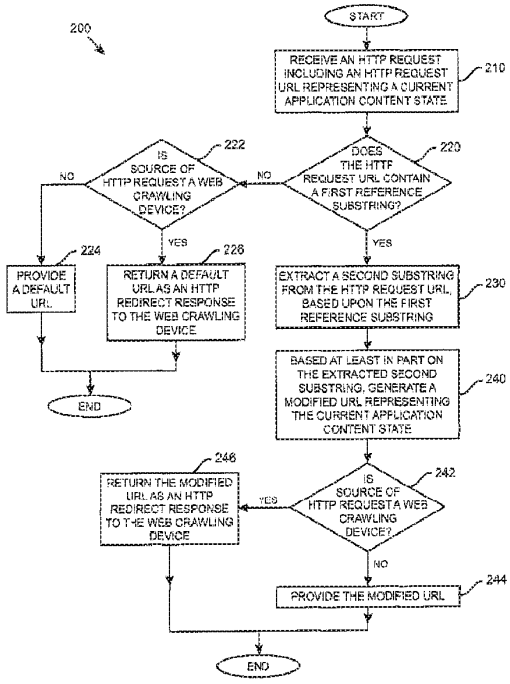
(72) Inventeurs/Inventors:
SCODA, ENRICO, IT;
PEZZANO, SIMONE, IT

(73) Propriétaire/Owner:
USABLENET INC., US

(74) Agent: PARLEE MCLAWS LLP

(54) Titre : PROCEDES PERMETTANT A DES APPLICATIONS WEB AJAX D'ETRE MISES EN SIGNETS ET D'ETRE PARCOURUES, ET DISPOSITIFS ASSOCIES

(54) Title: METHODS FOR MAKING AJAX WEB APPLICATIONS BOOKMARKABLE AND CRAWLABLE AND DEVICES THEREOF



(57) Abrégé/Abstract:
Methods, devices, and computer-readable storage media that make an asynchronous JavaScript and extensible mark-up language (AJAX) application content state bookmarkable and crawlable include receiving a HyperText Transfer Protocol (HTTP)

CA 2839006 C 2017/08/01
(11)(21) 2 839 006
(13) C

(57) Abrégé(suite)/Abstract(continued):
request is received. The HTTP request includes an HTTP request uniform resource locator (URL) representing a current application state. When the HTTP request URL contains a first reference substring, a second substring is extracted from the HTTP request URL based upon the first reference substring. A modified URL is generated based at least in part on the extracted second substring. The modified URL represents the current application content state.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
27 December 2012 (27.12.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/178167 A3

(51) International Patent Classification:
G06F 17/00 (2006.01) *G06F 15/16* (2006.01)
G06F 17/20 (2006.01)

(21) International Application Number:
PCT/US2012/044012

(22) International Filing Date:
25 June 2012 (25.06.2012)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
13/135,112 24 June 2011 (24.06.2011) US

(71) Applicant *(for all designated States except US)*: **US-ABLENET INC.** [US/US]; 28 W. 23rd Street, 6th Floor, New York, NY 10010 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **SCODA, Enrico** [IT/IT]; Via Cividina 416/3, I-33100 Martignacco (UD) (IT). **PEZZANO, Simone** [IT/IT]; Via Superiore, 21, I-33100 Udine (IT).

(74) Agents: **GALLO, nicolas J.** et al.; LeCLairRyan, 70 Linden Oaks Suite 210, Rochester, NY 14625 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

[Continued on next page]

(54) Title: METHODS FOR MAKING AJAX WEB APPLICATIONS BOOKMARKABLE AND CRAWLABLE AND DEVICES THEREOF

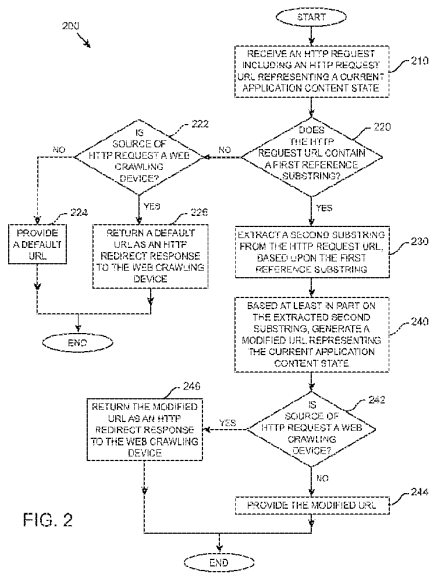


FIG. 2

(57) Abstract: Methods, devices, and computer-readable storage media that make an asynchronous JavaScript and extensible mark-up language (AJAX) application content state bookmarkable and crawlable include receiving a HyperText Transfer Protocol (HTTP) request is received. The HTTP request includes an HTTP request uniform resource locator (URL) representing a current application state. When the HTTP request URL contains a first reference substring, a second substring is extracted from the HTTP request URL based upon the first reference substring. A modified URL is generated based at least in part on the extracted second substring. The modified URL represents the current application content state.

WO 2012/178167 A3

WO 2012/178167 A3

(88) Date of publication of the international search report:
28 February 2013

METHODS FOR MAKING AJAX WEB APPLICATIONS BOOKMARKABLE AND CRAWLABLE AND DEVICES THEREOF

FIELD

[0001] This technology generally relates to methods for processing web application content and, more particularly, to methods for making an asynchronous JavaScript and extensible mark-up language (AJAX) application content state bookmarkable and crawlable and devices thereof.

BACKGROUND

[0002] An asynchronous JavaScript and extensible mark-up language (AJAX) web application uses AJAX to communicate with a server to load data or page fragments. Programming with AJAX offers benefits in the context of web application development. For example, AJAX offers site developers an additional level of innovation that may make a site easier to use for all visitors. AJAX may also be utilized to give users the sense that a web application is more responsive than a traditional web site.

[0003] However, with the use of AJAX to manipulate web page content, an issue arises with regard to providing bookmarkability of content that may be dynamic rather than static during a user session. For example, while a web page may have a single non-changing uniform resource locator (URL) stored in the web browser, the content in many AJAX web sites is asynchronously changed and replaced. Accordingly, when a user sets a bookmark, he may only be able to capture the starting URL of the web site instead of the exact content state he is currently viewing. Users therefore will want a way to bookmark a dynamic web page in a way that allows them to restore a specific content state.

[0004] One common solution is to change the hash part of the URL, since doing so does not activate any actions, such as loading a new page on the browser side. However, this approach yields yet another issue as to how a web crawler or

web robot can access that URL. Even if the hash part of the URL contains all of the information needed to fetch correct data using AJAX technology, the web crawler will not be able to access it since web crawlers are not able to execute JavaScript code inside the web pages they download.

SUMMARY

[0005] An exemplary method for making an asynchronous JavaScript and extensible mark-up language (AJAX) application content state bookmarkable and crawlable includes receiving at a web application processing server device a HyperText Transfer Protocol (HTTP) request including an HTTP request uniform resource locator (URL) representing a current application content state. When the HTTP request URL contains a first reference substring, a second substring based upon the first reference substring is extracted from the HTTP request URL with the web application processing server device. Based at least in part on the extracted second substring, a modified URL representing the current application content state is generated by the web application processing server device.

[0006] An exemplary computer-readable storage medium having stored thereon instructions for making an AJAX application content state bookmarkable and crawlable. The instructions include machine executable code which, when executed by at least one processor, causes the processor to perform steps including receiving a HTTP request including an HTTP request uniform resource locator (URL) representing a current application content state. When the HTTP request URL contains a first reference substring, a second substring based upon the first reference substring is extracted from the HTTP request URL. Based at least in part on the extracted second substring, a modified URL representing the current application content state is generated.

[0007] An exemplary web application processing server device including one or more processors and a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory, the programmed instructions including receiving a HTTP request including an HTTP

request uniform resource locator (URL) representing a current application content state. When the HTTP request URL contains a first reference substring, a second substring based upon the first reference substring is extracted from the HTTP request URL. Based at least in part on the extracted second substring, a modified URL representing the current application content state is generated.

[0008] This technology provides effective and efficient methods and devices for creating a concise URL representing a specific AJAX web application page or content state, which thereby allows generic web crawlers or robots to index the page or content state thus represented.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] FIG. 1 is a block diagram of a network environment with an exemplary web application processing server device; and

[0010] FIG. 2 is a flowchart of an exemplary method for making an AJAX application content state bookmarkable and crawlable.

DETAILED DESCRIPTION

[0011] A network environment 10 with an exemplary web application processing server device 12 is illustrated in FIG. 1. The environment 10 may include the web application processing server device 12, client computing devices 14(1)-14(n), web crawling devices 40(1)-40(n), a web server device 16, and communication networks 18(1)-18(2), although other numbers and types of systems, devices, servers, and/or elements in other configurations can be used. This technology provides effective and efficient methods and devices for creating a concise URL representing a specific AJAX web application page or content state, which thereby allows generic web crawlers or robots to index the page or content state thus represented.

[0012] This technology is directed to methods, devices, and computer-readable storage media for making an asynchronous JavaScript and extensible mark-up language (AJAX) application content state bookmarkable and crawlable.

In this example, the web application processing server device 12 receives a HyperText Transfer Protocol (HTTP) request including an HTTP request uniform resource locator (URL) representing a current application content state; extracts, from the HTTP request URL, a second substring based upon a first reference substring, when the HTTP request URL contains the first reference substring; and generates a modified URL based at least in part on the extracted second substring, wherein the modified URL represents the current application content state. Although one web application processing server device 12 is shown, other numbers and types of web application processing server systems can be used.

[0013] In some examples, the web application processing server device 12 may be a web content proxy server, or any other type of proxy server. For example, the web application processing server device 12 may manage the handling of redirect messages or redirection responses from the one or more web server devices 16 for and/or on behalf of requesting client devices 14(1)-14(n) and/or requesting web crawling devices 40(1)-40(n), although the web application processing server device 12 may provide other numbers and types of functions.

[0014] In other examples, the web application processing server device 12 may itself be a web server device. For example, the web application processing server device 12 may be contained in or identical to web server device 16. In such examples, the web application processing server device 12 may host one or more AJAX web applications and store associated web content in memory 15.

[0015] Referring more specifically to FIG. 1, the web application processing server device 12 may include a central processing unit (CPU) or processor 13, a memory 15, and an interface system 17 which are coupled together by a bus 19 or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor 13 in the web application processing server device 12 may execute a program of stored instructions one or more aspects of the present technology as described and illustrated by way of the examples herein, including methods for making an asynchronous AJAX application content state bookmarkable and

crawlable, although the processor could execute other numbers and types of programmed instructions.

[0016] The memory 15 in the web application processing server device 12 may store these programmed instructions for one or more aspects of the present technology as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor 13, may be used for the memory 15 in the web application processing server device 12. In some of these examples, the memory 15 may include a core module 21, a URL construction module 23, and a memory cache 25 which may store programmed instructions for one or more aspects of the present technology as described and illustrated herein, although the memory may comprise other types and numbers of systems, devices, and elements in other configurations which may store other data.

[0017] The interface device 17 in the web application processing server device 12 may be used to operatively couple and communicate between the web application processing server device 12 and the client computing devices 14(1)-14(n), web crawling devices 40(1)-40(n), and the web server device 16 via the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used. By way of example only, the communication networks 18(1) and 18(2) may use TCP/IP over Ethernet and industry-standard protocols, including HTTP, HTTPS, WAP, and SOAP, although other types and numbers of communication networks, such as a direct connection, a local area network, a wide area network, modems and phone lines, e-mail, and wireless and hardwire communication technology, each having their own communications protocols, may be used.

[0018] Each of the client computing devices 14(1)-14(n) may enable a user to access content and utilize one or more applications from the web server device 16 through the web application processing server device 12 through one or more communication networks, although one or more of the client computing devices 14(1)-14(n) could access content and utilize other types and numbers of applications from other sources and could provide a wide variety of other functions for the user. Although multiple client computing devices 14(1)-14(n) are shown, other numbers and types of user computing systems could be used. In this example, the client computing devices 14(1)-14(n) comprise devices with Internet access that permit a website page or other retrieved data to be displayed. By way of example only, one or more of the client computing devices 14(1)-14(n) may comprise smart phones, personal digital assistants, or desktop computers using visual or voice browsers.

[0019] Each of client computing devices 14(1)-14(n) may include a central processing unit (CPU) or processor 20, a memory 22, user input device 24, a display 26, and an interface system 28, and which are coupled together by a bus 30 or other link, although one or more of client computing devices 14(1)-14(n) may include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor 20 in each of client computing devices 14(1)-14(n) may execute a program of stored instructions for one or more aspects of the present technology as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions.

[0020] The memory 22 in each of the client computing devices 14(1)-14(n) may store these programmed instructions for one or more aspects of the present technology as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to

processor 20 may be used for the memory 22 in each of the client computing devices 14(1)-14(n).

[0021] The user input device 24 in each of the client computing devices 14(1)-14(n) may be used to input selections, such as requests for a particular website page, although the user input device could be used to input other types of data and interact with other elements. The user input device may include keypads, touch screens, and/or vocal input processing systems although other types and numbers of user input devices can be used.

[0022] The display 26 in each of the client computing devices 14(1)-14(n) may be used to show data and information to the user, such as a website page by way of example only. The display in each of the client computing devices 14(1)-14(n) may be a phone screen display, although other types and numbers of displays could be used.

[0023] The interface system 28 in each of the client computing devices 14(1)-14(n) may be used to operatively couple and communicate between the client computing devices 14(1)-14(n) and the web application processing server device 12 and web server device 16 over the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations may be used.

[0024] The web server device 16 may provide one or more web software applications for use by one or more of the client computing devices 14(1)-14(n), although the web server device 16 can provide other numbers and types of applications and/or content and can have provide other numbers and types of functions. Although one web server device 16 is shown for ease of illustration and discussion, other numbers and types of web server systems and devices can be used.

[0025] The web server device 16 may include a central processing unit (CPU) or processor, a memory, and an interface system which are coupled together by a bus or other link, although the web server device 16 may have other numbers and

types of components, parts, devices, systems, and elements in other configurations and locations may be used. The processor in the web server device 16 may execute a program of stored instructions for one or more aspects of the present technology as described and illustrated by way of the examples herein, including managing application functionality, although the processor could execute other numbers and types of programmed instructions.

[0026] The memory in the web server device 16 may store these programmed instructions for one or more aspects of the present technology as described and illustrated by way of the examples herein including managing application functionality, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor, may be used for the memory in each of the web server device 16.

[0027] The interface system in the web server device 16 may be used to operatively couple and communicate between the web server device 16 and the web application processing server device 12, the client computing devices 14(1)-14(n), and web crawling devices 40(1)-40(n) via communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[0028] Although examples of the web application processing server device 12, the client computing devices 14(1)-14(n), web crawling devices 40(1)-40(n), and the web server device 16, are described and illustrated herein, each of the client computing devices 14(1)-14(n), web crawling devices 40(1)-40(n), the web application processing server device 12, and the web server device 16, may be implemented on any suitable computer system or computing device. It is to be understood that the devices and systems of the examples described herein are for exemplary purposes, as many variations of the specific hardware and software

used to implement the examples are possible, as will be appreciated by those skilled in the relevant art(s).

[0029] Furthermore, each of the systems of the examples may be conveniently implemented using one or more general purpose computer systems, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the examples, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[0030] In addition, two or more computing systems or devices can be substituted for any one of the systems in any embodiment of the examples. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices and systems of the examples. The examples may also be implemented on computer system or systems that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[0031] The examples may also be embodied as a computer readable medium having instructions stored thereon for one or more aspects of the present technology as described and illustrated by way of the examples herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the examples, as described and illustrated herein.

[0032] Referring to FIGS. 1-2, an exemplary method 200 for making an AJAX application content state bookmarkable and crawlable will now be described. In step 210, the web application processing server device 12 receives

an HTTP request which includes an HTTP request URL representing a current application content state. In other examples, such as where the HTTP request originated from a client computing device 14, the HTTP request URL may correspond to a bookmark.

[0033] In step 220, the web application processing server device 12 determines whether the HTTP request contains a first reference substring. If in step 220 the HTTP request URL contains the first reference substring in step 220, then the Yes branch is taken to step 230.

[0034] In step 230, the web application processing server device 12 extracts a second substring from the HTTP request URL based upon the first reference substring. In some examples, the second substring may follow the first reference substring. In some of these examples, the second substring may include all of the text immediately following the first reference substring.

[0035] In step 240, the web application processing server device 12 generates a modified URL representing the current application content state based at least in part on the extracted second substring. The modified URL may, for example, be generated by the URL construction module 23 of the web application processing server device 12.

[0036] In step 242, the web application processing server device 12 determines if the source of the HTTP request is one of the web crawling device 40(1)-40(n) based upon a header in the HTTP request, although other manners for determining an identity of the source can be used. If in step 242 the web application processing server device 12 determines the source of the HTTP request is not one of the web crawling device 40(1)-40(n), then the No branch is taken to step 244. In step 244, the web application processing server device 12 provides the modified URL to the appropriate web server device 16. Based on the modified URL, the web server device 16 generates a response to the HTTP request which is then communicated to the source of the request (e.g., one of the client computing devices 14(1)-14(n)), and then the method ends. In some

embodiments, the response generated by the web server device 16 may include the current application content state corresponding to the modified URL.

[0037] If in step 242 the web application processing server device 12 determines the source of the HTTP request is one of the web crawling devices 40(1)-40(n), then the Yes branch is taken to step 246. In step 246, the web application processing server device 12 returns the modified URL as an HTTP redirect response to the identified one of the one of the web crawling device 40(1)-40(n). The web application processing server device 12 also may store the default URL at a location (e.g., in memory 15) associated with the HTTP redirect response. After step 246, this method ends.

[0038] If back in step 220, the received HTTP request URL does not contain the first reference substring in step 220, then the No branch is taken to step 222.

[0039] In step 222, the web application processing server device 12 determines based upon a header in the HTTP request if the source of the HTTP request is one of the web crawling devices 40(1)-40(n), although other manners for determining the source can be used. If in step 222 the web application processing server device 12 determines the source of the HTTP request is not one of the web crawling devices 40(1)-40(n), then the No branch is taken to step 224. In step 224, the web application processing server device 12 provides a default URL to the appropriate web server device 16. Based on the default URL, the web server device 16 generates a response to the HTTP request which is then communicated to the requesting one of the client computing devices 14(1)-14(n), and then the method ends, although other types of devices and systems could be the source of the request. In some embodiments, the response generated by the web server device 16 may include the current application content state corresponding to the default URL.

[0040] If in step 222 the web application processing server device 12 determines the source of the HTTP request is one of the web crawling devices 40(1)-40(n), then the Yes branch is taken to step 226. In step 226, the web

application processing server device 12 returns the default URL as an HTTP redirect response to the identified one of the web crawling devices 40(1)-40(n) that was the source of the HTTP request. The web application processing server device 12 also may store the default URL at a location (e.g., in memory 15) associated with the HTTP redirect response. After step 226 this method ends.

[0041] In an embodiment utilizing the exemplary method 200, a web application located at www.example.com/index.html may load the fragment welcome.html when starting up. This web application may also load product description pages identified by such links as /products/pdp.jsp?id=2234. The exemplary method 200 also may be utilized to address the problem of creating a URL representing an AJAX web application page or content state. One concept associated with exemplary method 200 is that relevant web pages (e.g., pages whose URLs will likely be saved or shared) may be a small portion of the entire web application. Moreover, such relevant web pages may often be identified by a small set of identifiers corresponding to keys in catalog databases or specific events (e.g., dates, flight numbers, etc.). For example, the URL /products/pdp.jsp?id=2234 may be represented by the identifier "2234". Therefore, in some examples, the AJAX web application JavaScript code may contain a function that creates a mapping between such an identifier and its corresponding URL. In some of these examples, a standard filename may be established for the portion of the JavaScript code containing this mapping function. The resulting JavaScript file containing this function may be limited to this mapping function and its dependencies (e.g., data structures or utility functions). Thus, a simple implementation of the JavaScript interpreter may be hosted on the web application processing server device 12, and the mapping function may be executed to obtain the URL being used by the web application AJAX engine. For example, in step 240, URL construction module 23 of web application processing server device 12 may, upon receiving such an identifier or second substring as an input (e.g., "2234"), generate the corresponding modified or processed URL associated with that identifier.

[0042] In another embodiment utilizing the exemplary method 200, an HTTP request URL may contain the first reference substring "/shareCode-" in step 220. In some of these examples, the substring following "/shareCode-" may be extracted in step 230 and passed along to the URL construction module 23 of the web application processing server device 12. In step 240, URL construction module 23 may then map this extracted second substring or identifier to a corresponding modified URL representing the current application content state. If, on the other hand, the HTTP request URL does not contain the first reference substring "/shareCode-" in step 220, then a default URL (e.g., a default startup URL) may be provided to the source of the HTTP request, as described in step 224 or step 226.

[0043] For example, consider an embodiment in which the HTTP request URL contains the path "/shareCode-2234". In step 230, the web application processing server device 12 extracts the identifier or second substring "2234" and passes it to the URL construction module 23, which returns the URL "/products/pdp.jsp?id=2234" in step 240. If, on the other hand, the URL path is equal to "/index.html", a default URL (e.g., "/welcome.html") may be returned in step 224 or step 226.

[0044] Beneficially, this approach results in short and concise URLs representing web application pages. Moreover, relevant data is in the URL path component such that it is universally recognized as a URL portion to keep. (Some web crawler devices, for example, will remove the hash portion of the URL when indexing.) Additionally, since only key information from the URL used by the AJAX call is used in the URL representing the web application content state, any marginal changes to the AJAX call URL will not invalidate the web application URL being saved, shared, or indexed.

[0045] In another embodiment utilizing exemplary method 200, one of the web crawling devices 40(1)-40(n) may send an HTTP request to the web application processing server device 12. The web application processing server device 12 may recognize that the HTTP request originated from one of the web

crawling devices 40(1)-40(n) by analyzing a header (e.g., the User-Agent HTTP request header) in the HTTP request. In some examples, if the header value is included in a list of supported or recognized web crawling devices, exemplary method 200 will be executed.

[0046] In some of these examples, if, in step 220, the HTTP request URL does not contain the first reference substring (e.g., "/shareCode-"), the web application processing server device 12 may provide to the one of the web crawling devices 40(1)-40(n) an HTTP redirect response containing the web application main HTML file (or any other suitable default URL) as content in step 226.

[0047] Otherwise, if the HTTP request URL does contain the first reference substring (e.g., "/shareCode-") in step 220, then, in some examples, the substring following "/shareCode-" may be extracted in step 230 and passed to the URL construction module 23. In steps 240, 242, and 246, the URL construction module 23 may generate and return the corresponding modified URL as an HTTP redirect response to one of the web crawling devices 40(1)-40(n). In this way, the one of the web crawling devices 40(1)-40(n) may access the URL of the HTML page containing the relevant information to be indexed.

[0048] Beneficially, this approach allows any of the web crawling devices 40(1)-40(n) to access AJAX web application pages or content states without a need to execute JavaScript code to emulate the AJAX web application behavior. Additionally, there is no need to add new standards or protocols to instruct web server devices and/or web crawling devices how to share information about AJAX web application pages.

[0049] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are

within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for making an asynchronous JavaScript and extensible mark-up language (AJAX) application content state bookmarkable and crawlable, the method comprising:

receiving at a web application processing server device a HyperText Transfer Protocol (HTTP) request comprising an HTTP request uniform resource locator (URL);

extracting from the HTTP request URL with the web application processing server device a second substring based upon a first reference substring, when the HTTP request URL contains the first reference substring;

generating by the web application processing server device a modified URL based at least in part on the extracted second substring, the modified URL representing a current application content state;

identifying, with the web application processing server device, when the HTTP request is from a web crawler, based upon a header in the HTTP request; and

returning with the web application processing server device the modified URL as an HTTP redirect response to the web crawler when the HTTP request is identified as being from the web crawler.

2. The method as set forth in claim 1 further comprising:

providing with the web application processing server device a default URL when the HTTP request URL does not contain the first reference substring.

3. The method as set forth in claim 1 wherein the second substring follows the first reference substring.

4. The method as set forth in claim 1 wherein the HTTP request URL corresponds to a bookmark.

5. The method as set forth in claim 1 further comprising:

storing by the web application processing server device the modified URL at a location associated with the HTTP redirect response.

6. A non-transitory computer readable medium having stored thereon instructions for making an asynchronous JavaScript and extensible mark-up language (AJAX) application content state bookmarkable or crawlable comprising machine executable code which when executed by at least one processor, causes the processor to perform steps comprising:

receiving a HyperText Transfer Protocol (HTTP) request comprising an HTTP request uniform resource locator (URL);

extracting, from the HTTP request URL, a second substring based upon a first reference substring, when the HTTP request URL contains the first reference substring;

generating a modified URL based at least in part on the extracted second substring, the modified URL representing a current application content state;

identifying when the HTTP request is from a web crawler, based upon a header in the HTTP request; and

returning the modified URL as an HTTP redirect response to the web crawler when the HTTP request is identified as being from the web crawler.

7. The medium as set forth in claim 6 further comprising:

providing a default URL when the HTTP request URL does not contain the first reference substring.

8. The medium as set forth in claim 6 wherein the second substring follows the first reference substring.

9. The medium as set forth in claim 6 wherein the HTTP request URL corresponds to a bookmark.

10. The medium as set forth in claim 6 further comprising:

storing the modified URL at a location associated with the HTTP redirect response.

11. A web application processing server device for making an asynchronous JavaScript and extensible mark-up language (AJAX) application content state bookmarkable and crawlable, the device comprising:

one or more processors;

a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory, the programmed instructions comprising:

receiving a HyperText Transfer Protocol (HTTP) request comprising an HTTP request uniform resource locator (URL);

extracting, from the HTTP request URL, a second substring based upon a first reference substring, when the HTTP request URL contains the first reference substring;

generating a modified URL based at least in part on the extracted second substring, the modified URL representing a current application content state;

identifying when the HTTP request is from a web crawler, based upon a header in the HTTP request; and

returning the modified URL as an HTTP redirect response to the web crawler when the HTTP request is identified as being from the web crawler.

12. The device as set forth in claim 11 further comprising:

providing a default URL when the HTTP request URL does not contain the first reference substring.

13. The device as set forth in claim 11 wherein the second substring follows the first reference substring.

14. The device as set forth in claim 11 wherein the HTTP request URL corresponds to a bookmark.

15. The device as set forth in claim 11 further comprising:

storing the modified URL at a location associated with the HTTP redirect response.

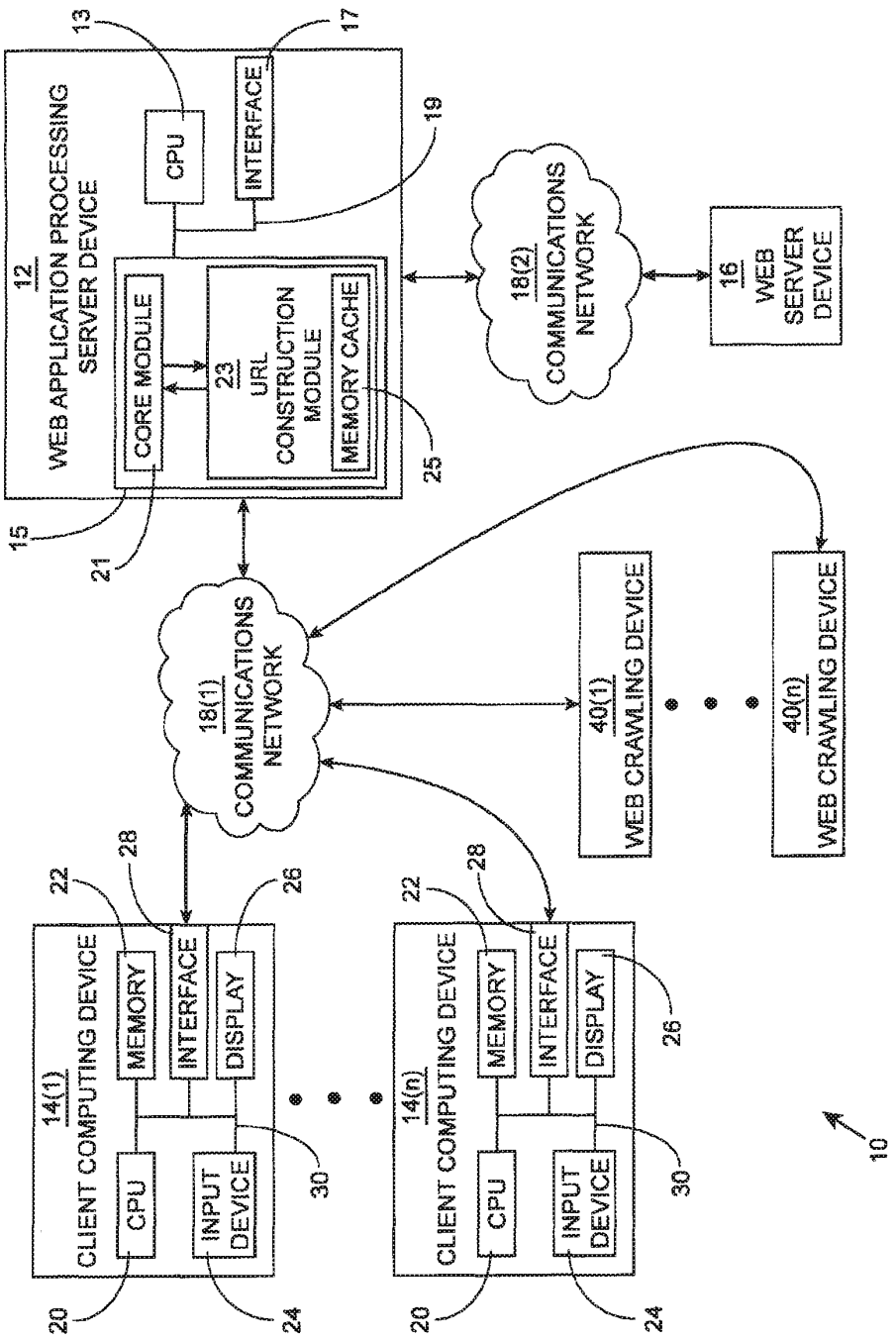

12
WEB APPLICATION PROCESSING SERVER DEVICE
13
CPU
17
INTERFACE
19
CORE MODULE
23
URL CONSTRUCTION MODULE
MEMORY CACHE
25
15
21
18(2)
COMMUNICATIONS NETWORK
16
WEB SERVER DEVICE
18(1)
COMMUNICATIONS NETWORK
40(1)
WEB CRAWLING DEVICE
40(n)
WEB CRAWLING DEVICE
14(1)
CLIENT COMPUTING DEVICE
MEMORY
INTERFACE
DISPLAY
CPU
INPUT DEVICE
22
28
26
30
20
24
14(n)
CLIENT COMPUTING DEVICE
MEMORY
INTERFACE
DISPLAY
CPU
INPUT DEVICE
22
28
26
30
20
24
10

FIG 1

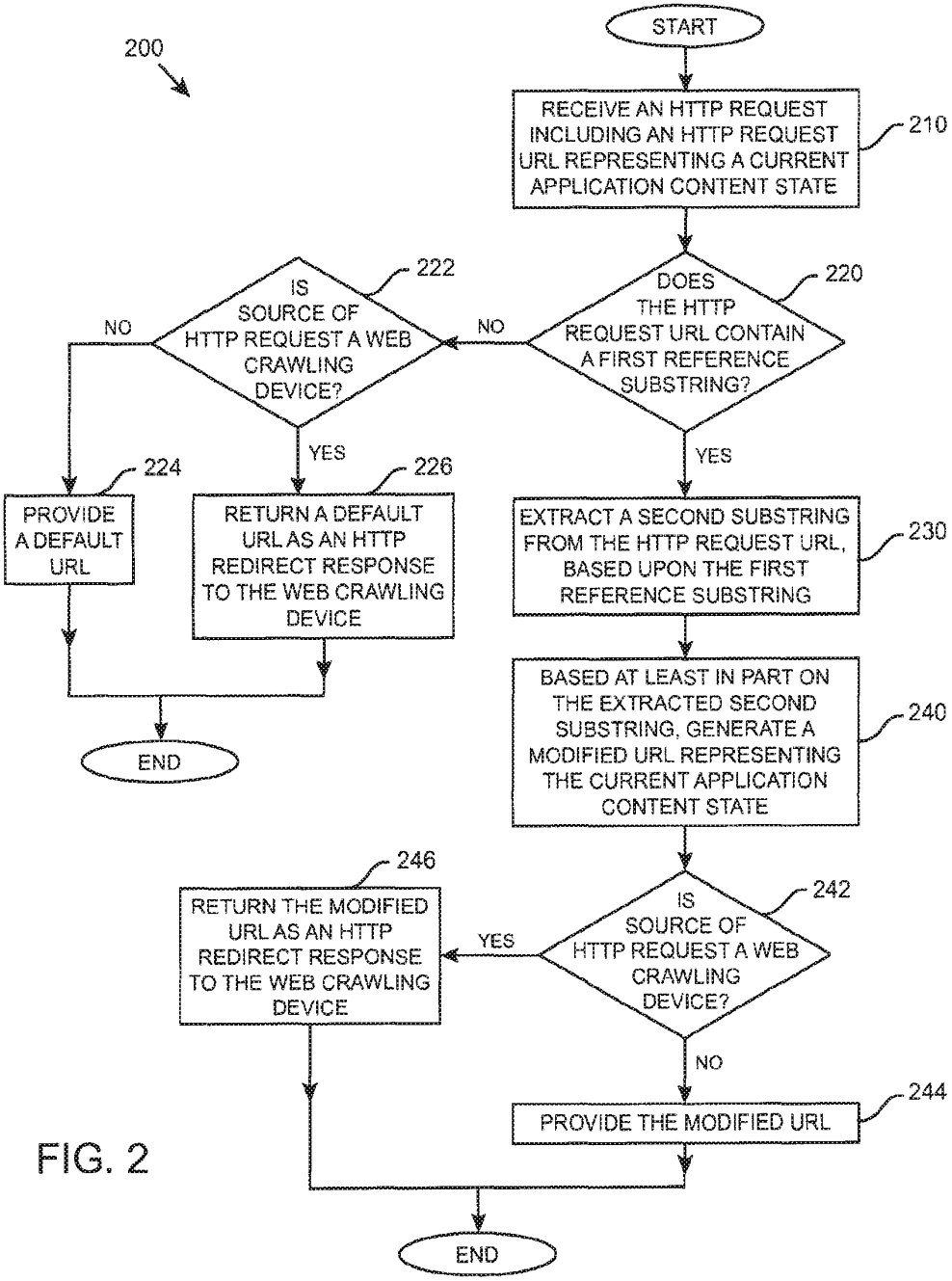
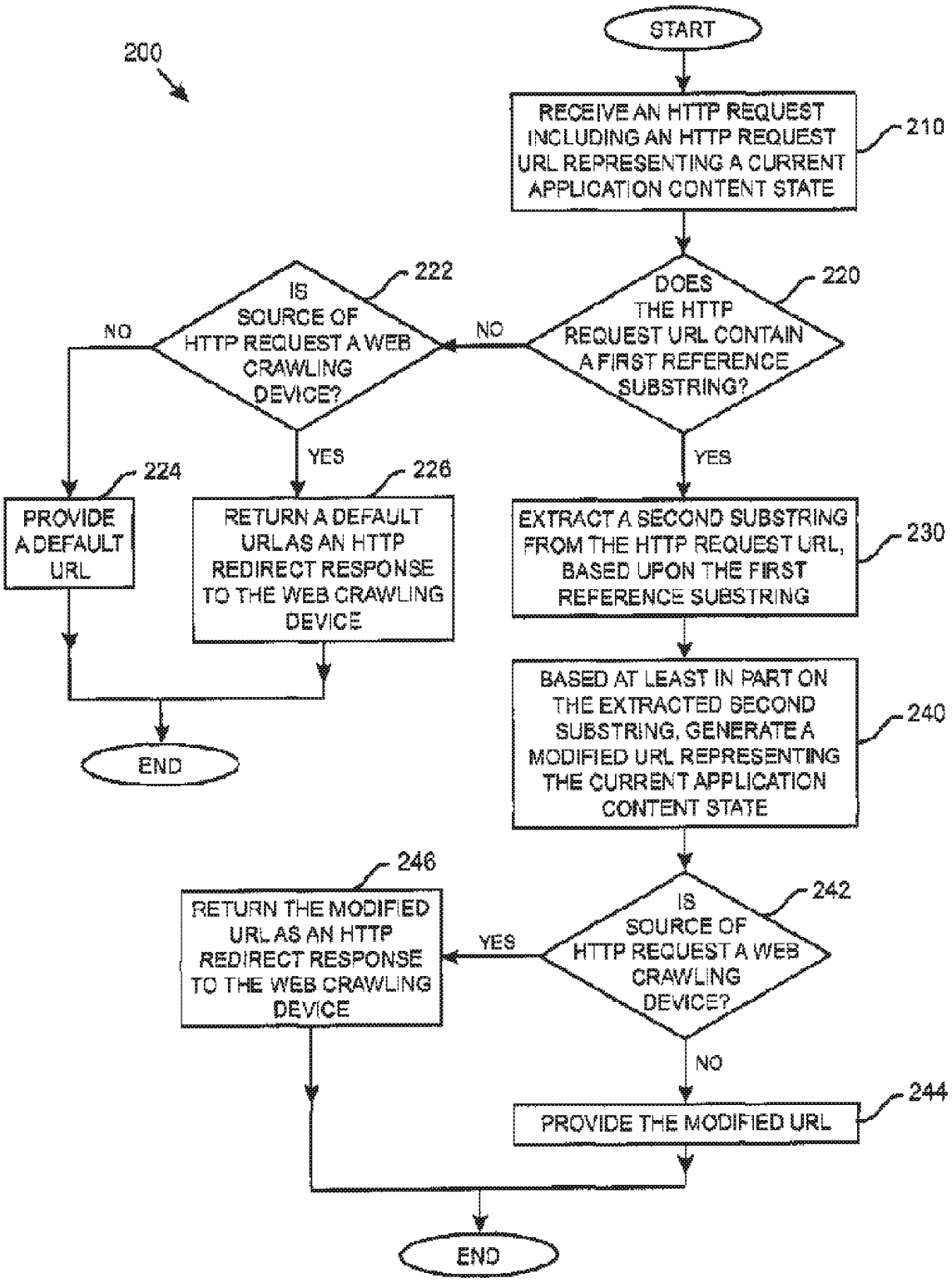

200
START
RECEIVE AN HTTP REQUEST INCLUDING AN HTTP REQUEST URL REPRESENTING A CURRENT APPLICATION CONTENT STATE
210
DOES THE HTTP REQUEST URL CONTAIN A FIRST REFERENCE SUBSTRING?
220
NO
IS SOURCE OF HTTP REQUEST A WEB CRAWLING DEVICE?
222
NO
YES
PROVIDE A DEFAULT URL
224
RETURN A DEFAULT URL AS AN HTTP REDIRECT RESPONSE TO THE WEB CRAWLING DEVICE
226
END
YES
EXTRACT A SECOND SUBSTRING FROM THE HTTP REQUEST URL, BASED UPON THE FIRST REFERENCE SUBSTRING
230
BASED AT LEAST IN PART ON THE EXTRACTED SECOND SUBSTRING, GENERATE A MODIFIED URL REPRESENTING THE CURRENT APPLICATION CONTENT STATE
240
IS SOURCE OF HTTP REQUEST A WEB CRAWLING DEVICE?
242
YES
RETURN THE MODIFIED URL AS AN HTTP REDIRECT RESPONSE TO THE WEB CRAWLING DEVICE
246
NO
PROVIDE THE MODIFIED URL
244
END

FIG. 2


200
START
RECEIVE AN HTTP REQUEST INCLUDING AN HTTP REQUEST URL REPRESENTING A CURRENT APPLICATION CONTENT STATE
210
DOES THE HTTP REQUEST URL CONTAIN A FIRST REFERENCE SUBSTRING?
220
NO
IS SOURCE OF HTTP REQUEST A WEB CRAWLING DEVICE?
222
NO
YES
YES
PROVIDE A DEFAULT URL
224
RETURN A DEFAULT URL AS AN HTTP REDIRECT RESPONSE TO THE WEB CRAWLING DEVICE
226
END
EXTRACT A SECOND SUBSTRING FROM THE HTTP REQUEST URL, BASED UPON THE FIRST REFERENCE SUBSTRING
230
BASED AT LEAST IN PART ON THE EXTRACTED SECOND SUBSTRING, GENERATE A MODIFIED URL REPRESENTING THE CURRENT APPLICATION CONTENT STATE
240
IS SOURCE OF HTTP REQUEST A WEB CRAWLING DEVICE?
242
YES
RETURN THE MODIFIED URL AS AN HTTP REDIRECT RESPONSE TO THE WEB CRAWLING DEVICE
246
NO
PROVIDE THE MODIFIED URL
244
END

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2839013 A1 2013/01/03

(21) 2 839 013

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(86) **Date de dépôt PCT/PCT Filing Date:** 2012/06/27
(87) **Date publication PCT/PCT Publication Date:** 2013/01/03
(85) **Entrée phase nationale/National Entry:** 2013/12/10
(86) **N° demande PCT/PCT Application No.:** US 2012/044395
(87) **N° publication PCT/PCT Publication No.:** 2013/003455
(30) **Priorité/Priority:** 2011/06/27 (US61/501,391)

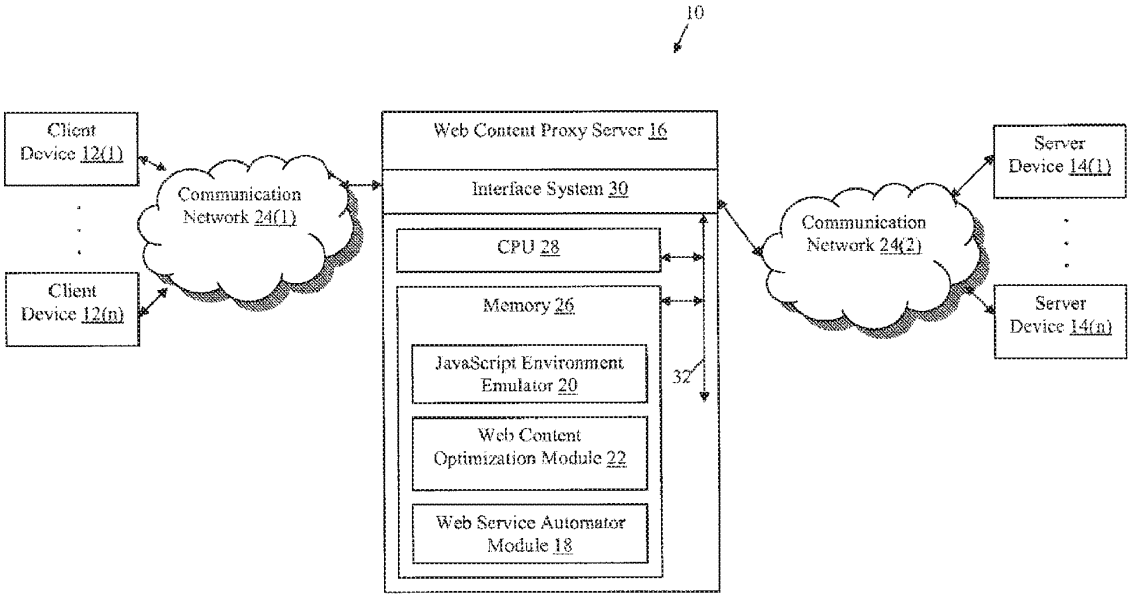
(51) **Cl.Int./Int.Cl.** ***G06F 17/00*** (2006.01),
G06F 15/16 (2006.01)

(71) **Demandeur/Applicant:**
USABLENET INC., US

(72) **Inventeurs/Inventors:**
SCODA, ENRICO, IT;
BRONDANI, MARCO, IT

(74) **Agent:** INGALLS, DORAN J.

(54) **Titre : PROCEDES POUR LA MISE EN OEUVRE DE SERVICES WEB ET DISPOSITIFS ASSOCIES**
(54) **Title: METHODS FOR IMPLEMENTING WEB SERVICES AND DEVICES THEREOF**



(57) **Abrégé/Abstract:**
A method, non-transitory computer readable medium, and apparatus that determines when a JavaScript file is associated with a received web service request. A JavaScript environment is emulated when the determining indicates the JavaScript file is associated with the received web service request. A JavaScript object including one or more parameters of the received web service request is created. The JavaScript file associated with the received web service request is executed in the emulated JavaScript environment using the the JavaScript object. A web service response is constructed based on the JavaScript object as populated based on the executing. The constructed web service response is provided to a source of the received web service request.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
3 January 2013 (03.01.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/003455 A3

(51) International Patent Classification:
G06F 17/00 (2006.01) *G06F 15/16* (2006.01)

(21) International Application Number:
PCT/US2012/044395

(22) International Filing Date:
27 June 2012 (27.06.2012)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/501,391 27 June 2011 (27.06.2011) US

(71) Applicant *(for all designated States except US)*: **US-ABLENET INC.** [US/US]; 28 W. 23rd St., 6th Floor, New York, NY 10010 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **SCODA, Enrico** [IT/IT]; Via Cividina 416/3, I-33035 Martignacco (ud) (IT). **BRONDANI, Marco** [IT/IT]; Via Paniae 24, 33010 Colloredo Di Monte Albano, Udine (IT).

(74) Agents: **GOLDMAN, Michael, L.** et al.; Leclairryan, A Professional Corporation, 70 Linden Oaks, Suite 210, Rochester, NY 14625 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

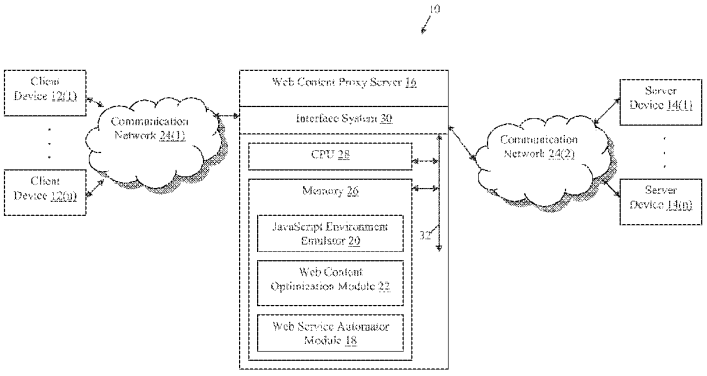
Published:

---- *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(88) Date of publication of the international search report:
14 March 2013

(54) Title: METHODS FOR IMPLEMENTING WEB SERVICES AND DEVICES THEREOF



(57) Abstract: A method, non-transitory computer readable medium, and apparatus that determines when a JavaScript file is associated with a received web service request. A JavaScript environment is emulated when the determining indicates the JavaScript file is associated with the received web service request. A JavaScript object including one or more parameters of the received web service request is created. The JavaScript file associated with the received web service request is executed in the emulated JavaScript environment using the the JavaScript object. A web service response is constructed based on the JavaScript object as populated based on the executing. The constructed web service response is provided to a source of the received web service request.

WO 2013/003455 A3

METHODS FOR IMPLEMENTING WEB SERVICES AND DEVICES THEREOF

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims the benefit of U.S. Provisional Patent Application Serial No. 61/501,391, filed June 27, 2011, which is hereby incorporated by reference in its entirety.

BACKGROUND

[0002] Web services provide a standardized way of integrating web-based applications using eXtensible Markup Language (XML), SOAP, Web Services Description Language (WSDL), and/or Universal Description Discovery and Integration (UDDI) standards over an Internet Protocol (IP) backbone. XML can be used to tag data used by a web service, SOAP can be used to transfer the data, WSDL can be used for describing the web services available and UDDI can be used for listing the available web services. Web services allow different applications located at different sources to communicate with each other efficiently and without custom coding which can require a significant amount of resources. Additionally, because communications are in XML, web services are not tied to any operating system or programming language.

[0003] Unlike traditional client/server models, web services do not provide an end user with a graphical user interface (GUI). Instead, web services share data and processes through an application interface across a network. These application interfaces are invoked and used to interpret any resulting data.

[0004] With the growing market for applications, web services are increasingly popular since it is easy to integrate them into applications to extend the features offered to the end user. Unfortunately, there are limitations on the ability of mobile computing devices to effectively utilize web services integrated into such applications including, for example, with respect to the execution of functionality implemented using JavaScript code.

SUMMARY

[0005] A method for implementing a web service includes determining with a web content proxy server when a JavaScript file is associated with a received web service request. A JavaScript environment is emulated with the web content proxy server when the determining indicates the JavaScript file is associated with the received web service request. A JavaScript object including one or more parameters of the received web service request is created with the web content proxy server. The JavaScript file associated with the received web service request is executed with the web content proxy server in the emulated JavaScript environment using the the JavaScript object. The emulated JavaScript environment is capable of facilitating execution of web transactions including one or more pages. A web service response is constructed with the web content proxy server based on the JavaScript object as populated based on the executing. The constructed web service response is provided with the web content proxy server to a source of the received web service request.

[0006] A non-transitory computer readable medium having stored thereon instructions for implementing a web service comprising machine executable code which when executed by at least one processor, causes the processor to perform steps including determining whether a JavaScript file is associated with a received web service request. A JavaScript environment is emulated when it is determined that there is a JavaScript file associated with the received web service request. A JavaScript object including one or more parameters of the received web service request is created. The JavaScript file associated with the received web service request is executed in the emulated JavaScript environment using the the JavaScript object. The emulated JavaScript environment is capable of facilitating execution of web transactions including one or more pages. A web service response is constructed based on the JavaScript object as populated based on the executing. The constructed web service response is provided to a source of the received web service request.

[0007] A web content proxy server includes one or more processors and a memory coupled to the one or more processors which are configured to execute

programmed instructions stored in the memory including determining whether a JavaScript file is associated with a received web service request. A JavaScript environment is emulated when it is determined that there is a JavaScript file associated with the received web service request. A JavaScript object including one or more parameters of the received web service request is created. The JavaScript file associated with the received web service request is executed in the emulated JavaScript environment using the the JavaScript object. The emulated JavaScript environment is capable of facilitating execution of web transactions including one or more pages. A web service response is constructed based on the JavaScript object as populated based on the executing. The constructed web service response is provided to a source of the received web service request.

[0008] This technology provides a number of advantages including providing methods and devices for more easily implementing web services with a web content proxy server executing a web service automator module and JavaScript environment emulator. With this technology, web services can be more effectively extended and utilized by mobile computing devices that may not otherwise be capable of exploiting such functionality.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] FIG. 1 is a block diagram of an environment with an exemplary web content proxy server with a web content optimization module, web service automator module, and a JavaScript environment emulator;

[0010] FIG. 2 is an exemplary login page, such as may be located at a *.login.html web address, and its hypertext markup language (HTML) source code;

[0011] FIG. 3 is an exemplary successful login page including a welcome message and a list of links to access different functionalities and logout and its HTML source code;

[0012] FIG. 4 is a flow chart of an exemplary method for implementing web services; and

[0013] FIG. 5 is exemplary JavaScript code for implementing a function facilitating generation of a web service response.

DETAILED DESCRIPTION

[0014] An exemplary environment 10 with client devices 12(1)-12(n), server devices 14(1)-14(n), and web content proxy server 16, including a web service automator module 18, a JavaScript environment emulator 20, and a web content optimization module 22, coupled to communication networks 24(1)-24(2) is illustrated in FIG. 1. Other numbers and types of systems, devices, and/or elements in other configurations and environments with other communication network topologies can also be used. By way of example only, the web service automator module 18, JavaScript environment emulator 20, and/or web content optimization module 22 can be implemented as an executable module of programmed instructions for one or more of the methods described herein which is stored in a memory 26 and executed by one or more processors or CPUs 28 in the web content proxy server 16. This technology provides a number of advantages including providing methods, non-transitory computer readable medium, and devices for more easily and effectively implementing web services on behalf of mobile computing devices.

[0015] Referring more specifically to FIG. 1, the web content proxy server 16 includes a central processing unit (CPU) 28 or processor, a memory 26, and an interface system 30 which are coupled together by a bus 32 or other link, although other numbers and types of components, parts, devices, systems, and elements in other configurations and locations can be used. The processor 28 in the web content proxy server 16 executes a program of stored instructions one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor 28 could execute other numbers and types of programmed instructions.

[0016] The memory 26 in the web content proxy server 16 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed

instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a random access memory (RAM) or a read only memory (ROM) in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor 28, can be used for the memory 26 in the web content proxy server 16. In some embodiments, the memory 26 includes a web service automator module 18, a JavaScript environment emulator 20, and a web content optimization module 22 comprising programmed instructions and/or configurable hardware logic for one or more aspects of the present invention as described and illustrated herein, although the memory 26 can comprise other types and numbers of systems, devices, and elements in other configurations which store other data.

[0017] The web service automator module 18 can include programmed instructions and/or logic configured to facilitate one or more of the methods described and illustrated in detail below including receiving a web service request from one of the client devices 12(1)-12(n) and providing a web service response. The JavaScript environment emulator 20 can include programmed instructions and/or logic configured to simulate a JavaScript environment for executing one or more JavaScript files, as described and illustrated in more detail below. The web content optimization module can include programmed instructions and/or logic configured to extract content relevant to a requesting client device 12(1)-12(n) and adapt, transform, or otherwise modify this extracted content to fit the specifications of the requesting client device 12(1)-12(n). One or more of the web service automator module 18, the JavaScript environment emulator 20, or the web content optimization module 22 can also have other types and numbers of functions as described and illustrated herein. Additionally, one or more of the web service automator module 18, the JavaScript environment emulator 20, or the web content optimization module 22 can be stored at and/or implemented by a separate device coupled to the web content proxy server 16 by one or more of the communication networks 24(1)-24(2), for example.

[0018] The interface system 30 in the web content proxy server 16 is used to operatively couple and communicate between the web content proxy server 16,

the client devices 12(1)-12(n) and the server devices 14(1)-14(n) via the communication networks 24(1)-24(2), although other types and numbers of communication networks 24(1)-24(2) with other types and numbers of connections and configurations can be used. Additionally, one or more of the communication networks 24(1)-24(2) can include one or more networks, such as one or more local area networks (LANs) and/or wide area networks (WANs). By way of example only, the communication networks can use TCP/IP over Ethernet and industry-standard protocols, including hypertext transfer protocol (HTTP), secure HTTP (HTTPS), wireless application protocol (WAP), and/or SOAP, although other types and numbers of communication networks, such as a direct connection, modems and phone lines, e-mail, and wireless and hardwire communication technology, each having their own communications protocols, can be used.

[0019] The client devices 12(1)-12(n) enable a user to request, receive, and interact with applications, web services, and content hosted by the server devices 14(1)-14(n) through the web content proxy server 16 via one or more communication networks 24(1), although one or more of the client devices 12(1)-12(n) could access content and utilize other types and numbers of applications from other sources and could provide a wide variety of other functions for the user. In some examples, the client devices 12(1)-12(n) comprise mobile computing devices with Internet access that enable one or more web services to be accessed. By way of example only, the client devices 12(1)-12(n) can be smart phones, personal digital assistants, or computers.

[0020] In some examples, the client devices 12(1)-12(n) include a central processing unit (CPU) or processor, a memory, a user input device, a display, and an interface system, which are coupled together by a bus or other link, although one or more of client devices 12(1)-12(n) can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor in the client devices 12(1)-12(n) can execute a program of instructions stored in the memory of the client device 12(1)-12(n) for one or more aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions.

[0021] The user input device in the client devices 12(1)-12(n) can be used to input selections, such as a request for a particular web service, although the user input device could be used to input other types of requests and data and interact with other elements. The user input device can include keypads, touch screens, and/or vocal input processing systems although other types and numbers of user input devices can be used.

[0022] The display the client device can be used to show data and information to the user, such as a response to the web service request by way of example only. The display in the client device is a phone screen display, although other types and numbers of displays could be used depending on the particular type of client device.

[0023] The interface system in the client devices 12(1)-12(n) can be used to operatively couple and communicate between the client devices 12(1)-12(n), the web content proxy server 16, and the server devices 14(1)-14(n) over the communication networks 24(1)-24(2).

[0024] The server devices 14(1)-14(n) provide content including web pages for use by one or more of the client devices 12(1)-12(n) via the web content proxy server 16, although the server devices 14(1)-14(n) can provide other numbers and types of functions.

[0025] Each of the server devices 14(1)-14(n) can include a central processing unit (CPU) or processor, a memory, and an interface system which are coupled together by a bus or other link, although each of the web server devices 14(1)-14(n) could have other numbers and types of components, parts, devices, systems, and elements in other configurations and locations. The processor in each of the server devices 14(1)-14(n) executes a program of instructions stored in the memory of the server devices 14(1)-14(n) for one or more aspects of the present invention as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[0026] The interface system in each of the server devices 14(1)-14(n) is used to operatively couple and communicate between the server devices 14(1)-14(n), the web content proxy server 16, and the client devices 12(1)-12(n) via communication networks 24(1)-24(2).

[0027] Although embodiments web content proxy server 16, the client devices 12(1)-12(n), and the server devices 14(1)-14(n), are described and illustrated herein, each of the web content proxy server 16, the client devices 12(1)-12(n), and the server devices 14(1)-14(n), can be implemented on any suitable computer apparatus or computing device. It is to be understood that the apparatuses and devices of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software used to implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s).

[0028] Furthermore, each of the devices of the embodiments may be conveniently implemented using one or more general purpose computers, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[0029] In addition, two or more computing apparatuses or devices can be substituted for any one of the devices in any embodiment described herein. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices of the embodiments. The embodiments may also be implemented on computer apparatuses or devices that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[0030] The embodiments may also be embodied as one or more non-transitory computer readable medium having instructions stored thereon for one or more aspects of the present invention as described and illustrated by way of the embodiments herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the embodiments, as described and illustrated herein.

[0031] An exemplary method for implementing web services will now be described below. This technology defines methods, non-transitory computer readable medium, and devices to implement web services using web site(s) and/or web page(s), and associated content and/or source code as the source data, to produce results including one or more web service responses. Additionally, as many web sites depend on web cookies as a way to store and keep the state of a transaction or session, web services implemented as described with this technology can extend the standard web service definition to include web cookies to keep the mapping between web services and a web site data source.

[0032] This technology also utilizes a JavaScript environment emulator 20 configured to make asynchronous JavaScript and XML (Ajax) calls and to execute web site transactions corresponding to the functionality implemented by web services. By way of example only, a JavaScript environment emulator 20 in a web content proxy server 16 is illustrated and described in U.S. Patent Application Serial No. 12/802,670 entitled, "Methods For Utilizing A JavaScript Emulator In A Web content proxy Server And Devices Thereof" which is herein incorporated by reference in its entirety.

[0033] As illustrated in FIG. 1, the web service automator module 18 can be an application comprising programmed instructions as described and illustrated herein and executed by the web content proxy server 16 that interacts with one of the client devices 12(1)-12(n) making a web service request and server device(s) 14(1)-14(n) acting as the source data for the web service, although other configurations can be used. By way of example, the web service automator module 18 can be a module in memory 26 executed by the web content proxy server 16. Optionally, the web content proxy server 16 can further execute the

web content optimization module 22 which can be used as a "preprocessor" that modifies web content before the content is passed to the web service automator module 18.

[0034] Referring to FIG. 2, a diagram of an exemplary login page 200, such as may be located at a *login.html uniform resource locator (URL) or web address, where "*" is a wild card referring to a host name and/or one or more higher level domains, for example, and its HTML source code 202, is shown. The HTML source code 202 includes a form pointing to a /login.do script and having a "sessionid" hidden field, "user" and "passwd" text fields for receiving inserted login credentials, and a submit button labeled "Login".

[0035] When a user of one of the client devices 12(1)-12(n) submits valid login credentials and engages the "Login" submit button on the web page 200 with a user input device, the user can receive a web page, such as the exemplary web page 300 shown in FIG. 3. The web page 300 includes a welcome message and a list of links to access different functionalities and logout.

[0036] Further illustrated in FIG. 3 is the HTML source code 302 associated with the successful login web page 300. Upon submission of the form, the submitted values for the "user" and "passwd" fields will be communicated along with the value for the hidden "sessionid" field. Optionally, the responding device, such as one of the server devices 14(1)-14(n), can use a set-cookie header populated with the "sessionid" field value (e.g. sessionid=dhfumsn3942b) as a cookie to store identification information for the session. Accordingly, in this example, an expected web service response for a submission including a valid username and password is XML data that can be rendered in a browser of the client device 12(1)-12(n) and which includes a welcome message for the validated user such as "<message>welcome message...</message>".

[0037] Referring to FIG. 4, an exemplary method for implementing a web service is illustrated. In step 400, the web content proxy server 16 executing the web service automator module 18 receives a web service request from one of the client devices 12(1)-12(n). Referring to a web service implementation of the login

process illustrated in FIGS. 2 and 3, the received web service request can be associated with a URL such as "https://processor.com/ws/samplesite/login?username=jake&password=123", where processor.com is the web site hosting the web service, /ws/ is a web application running a web service engine, samplesite/login is a URL substring identifying the web service, and the username and password are the two parameters of the web service. While the username and password are shown in this exemplary URL, the user name and password parameter values can be hidden using techniques that are well-known in the art.

[0038] In step 402, the web content proxy server 16 executing the web service automator module 18 determines whether there is a JavaScript file corresponding to the web service. Accordingly, in the above example, the web content proxy server 16 executing the web service automator module 18 determines whether there is a project in the memory called "samplesite" which contains a JavaScript file called "login.js". If the web content proxy server 16 executing the web service automator module 18 determines that the identified JavaScript file does not exist, then a No branch is taken to step 404. In step 404, the web content proxy server 16 executing the web service automator module 18 sends an error page web service response to the requesting one of the client devices 12(1)-12(n) indicating the web service is not available.

[0039] If the web content proxy server 16 executing the web service automator module 18 determines that the identified JavaScript file does exist, then a Yes branch is taken to step 406. In step 406, the web content proxy server 16 executing the web service automator module 18 communicates with the JavaScipt environment emulator 20 in order to generate a JavaScript environment, such as illustrated and described by way of example only in U.S. Patent Application Serial No. 12/802,670 which is herein incorporated by reference in its entirety.

[0040] In step 408, the web content proxy server 16 executing the web service automator module 18 creates a JavaScript object representing the web service, although the object can be generated in other manners. With reference to FIG. 5, in which JavaScript code 500 for implementing a function facilitating

generation of a web service response is shown, the JavaScript object can be named "webService", although other names can also be used. In this example, the created JavaScript object contains the parameters of the web service request (e.g. user name and password values), as retrieved from the URL and/or through HTTP GET and/or POST commands, for example.

[0041] In step 410, the web content proxy server 16 executing the JavaScript environment emulator 20 executes a function in the matching JavaScript file identified in step 402. In the example shown in FIG. 5, the function is named "service", although other functions and/or names, the identity of which is known by the web content proxy server 16, can also be used. The function is executed using the environment generated in step 406 and the created web service object as an input parameter. In this example, the "service" function uses a library, such as jQuery, that simplifies the use of Ajax inside the JavaScript environment in the JavaScript environment emulator 20 of the web content proxy server 16, although any other functions or libraries can also be used. The executed function is responsible for populating the created web service object (e.g. the "webService" object), with the necessary data to construct a web service response.

[0042] The exemplary "service" function shown in the JavaScript code of FIG. 5 retrieves the login.html page from the responding one of the server devices 14(1)-14(n). Next, the "service" function fills the login.html form text fields with the parameters passed to the function as contained by the web service object input parameter (e.g. "webService"). Once filled, the "service" function submits the form data to the "login.do" script indicated by the form action in the HTML source code 202, simulating submission of the form by the requesting one of the client devices 12(1)-12(n). The "login.do" script can, for example, validate the user name and password parameter values, although any other functionality can also be provided.

[0043] In step 412, the web content proxy server 16 executing the web service automator module 18 generates a web service response based on the input web service object, as modified by the executed function. Accordingly, in the

example shown in FIG. 5, the "service" function further sets a "response" property of the "webService" web service object to include "<message>" and "</message>" XML tags with a text message (e.g. "Hi Jake, welcome back!") inserted between the tags. Any other XML tags, or any other code capable of being interpreted by the web service automator module 18, and/or the browser of the requesting one of the client devices 12(1)-12(n), can be used to populate the web service object. Additionally, any other web service functionality can be provided by the JavaScript code 500. Accordingly, the web content proxy server 16 executing the web service automator module 18 inspects the web service object returned from, and as modified by, the JavaScript environment emulator 20, and generates the web service response.

[0044] In step 414, the web content proxy server 16 executing the web service automator module 18 provides the generated web service response to the requesting one of the client devices 12(1)-12(n). The response is in a form that can be rendered by the requesting one of the client devices 12(1)-12(n). Optionally, the web content proxy server 16 executing the web service automator module 18 includes a cookie, such as a cookie including the sessionid information in this example, in order to maintain the state of the session between the requesting one of client devices 12(1)-12(n) and the responding one of the server devices 14(1)-14(n).

[0045] Accordingly, as illustrated and described herein this technology provides a number of advantages including providing methods, non-transitory computer readable medium, and devices for more easily implementing web services with a web content proxy server. With this technology, a web content proxy server can implement a web service thereby allowing mobile computing devices to send and receive web service requests and responses even though the devices may not be capable of performing such functionality. Thereby, web services can be more effectively extended and utilized by mobile computing devices.

[0046] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is

intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations, improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for implementing a web service, the method comprising:

determining with a web content proxy server when a JavaScript file is associated with a received web service request;

emulating with the web content proxy server a JavaScript environment when the determining indicates the JavaScript file is associated with the received web service request;

creating with the web content proxy server a JavaScript object including one or more parameters of the received web service request;

executing with the web content proxy server the JavaScript file associated with the received web service request in the emulated JavaScript environment using the the JavaScript object;

constructing with the web content proxy server a web service response based on the JavaScript object as populated based on the executing; and

providing with the web content proxy server the constructed web service response to a source of the received web service request.

2. The method as set forth in claim 1 further comprising setting with the web content proxy server one or more cookies based on the constructed web service response.

3. The method as set forth in claim 1 further comprising providing with the web content proxy server an error message to the source of the received web service request as a web service response when there is no determined JavaScript file associated with the received web service request.

4. The method as set forth in claim 1 wherein the web content proxy server comprises at least one of a web service automator module, a web content optimization module, or a JavaScript environment emulator.

5. The method as set forth in claim 4 wherein the JavaScript environment emulator is configured to make asynchronous JavaScript and XML (Ajax) calls and execute web site transactions corresponding to functionality implemented by one or more web services.

6. The method as set forth in claim 1 wherein the recevied web service request comprises at least one of a GET HTTP request, a POST HTTP request, a PUT HTTP request, or a DELETE HTTP request.

7. The method as set forth in claim 1 wherein the received web service request and the constructed web service response contain cookie and set-cookie headers, respectively.

8. The method as set forth in claim 1 wherein the constructed web service response comprises one of a hypertext markup language (HTML), JavaScript Object Notation (JSON), or eXtensible Markup Language (XML) response.

9. A non-transitory computer readable medium having stored thereon instructions for implementing a web service comprising machine executable code which when executed by at least one processor, causes the processor to perform steps comprising:

determining when a JavaScript file is associated with a received web service request;

emulating a JavaScript environment when the determining indicates the JavaScript file is associated with the received web service request;

creating a JavaScript object including one or more parameters of the received web service request;

executing the JavaScript file associated with the received web service request in the emulated JavaScript environment using the the JavaScript object;

constructing a web service response based on the JavaScript object as populated based on the executing; and

providing the constructed web service response to a source of the received web service request.

10. The medium as set forth in claim 9 further having stored thereon machine executable code that when executed by the at least one processor cause the processor to perform steps further comprising setting one or more cookies based on the constructed web service response.

11. The medium as set forth in claim 9 further having stored thereon machine executable code that when executed by the at least one processor cause the processor to perform steps further comprising providing an error message to the source of the received web service request as a web service response when there is no determined JavaScript file associated with the received web service request.

12. The medium as set forth in claim 9 wherein the recevied web service request comprises at least one of a GET HTTP request, a POST HTTP request, a PUT HTTP request, or a DELETE HTTP request.

13. The medium as set forth in claim 9 wherein the received web service request and the constructed web service response contain cookie and set-cookie headers, respectively.

14. The medium as set forth in claim 9 wherein the constructed web service response comprises one of a hypertext markup language (HTML), JavaScript Object Notation (JSON), or eXtensible Markup Language (XML) response.

15. A web content proxy server, comprising:

one or more processors; and

a memory coupled to the one or more processors which are configured to execute programmed instructions stored in the memory comprising:

determining when a JavaScript file is associated with a received web service request;

emulating a JavaScript environment when the determining indicates the JavaScript file is associated with the received web service request;

creating a JavaScript object including one or more parameters of the received web service request;

executing the JavaScript file associated with the received web service request in the emulated JavaScript environment using the the JavaScript object;

constructing a web service response based on the JavaScript object as populated based on the executing; and

providing the constructed web service response to a source of the received web service request.

16. The apparatus as set forth in claim 15 wherein the one or more processors are further configured to execute programmed instructions stored in the memory further comprising setting one or more cookies based on the constructed web service response.

17. The apparatus as set forth in claim 15 wherein the one or more processors are further configured to execute programmed instructions stored in the memory further comprising providing with the web content proxy server an error message to the source of the received web service request as a web service response when there is no determined JavaScript file associated with the received web service request.

18. The apparatus as set forth in claim 15 further comprising at least one of a web service automator module, a web content optimization module, or a JavaScript environment emulator.

19. The apparatus as set forth in claim 18 wherein the JavaScript environment emulator is configured to make asynchronous JavaScript and XML (Ajax) calls and execute web site transactions corresponding to functionality implemented by one or more web services.

20. The apparatus as set forth in claim 15 wherein the recevied web service request comprises at least one of a GET HTTP request, a POST HTTP request, a PUT HTTP request, or a DELETE HTTP request.

21. The apparatus as set forth in claim 15 wherein the received web service request and the constructed web service response contain cookie and set-cookie headers, respectively.

22. The apparatus as set forth in claim 15 wherein the constructed web service response comprises one of a hypertext markup language (HTML), JavaScript Object Notation (JSON), or eXtensible Markup Language (XML) response.

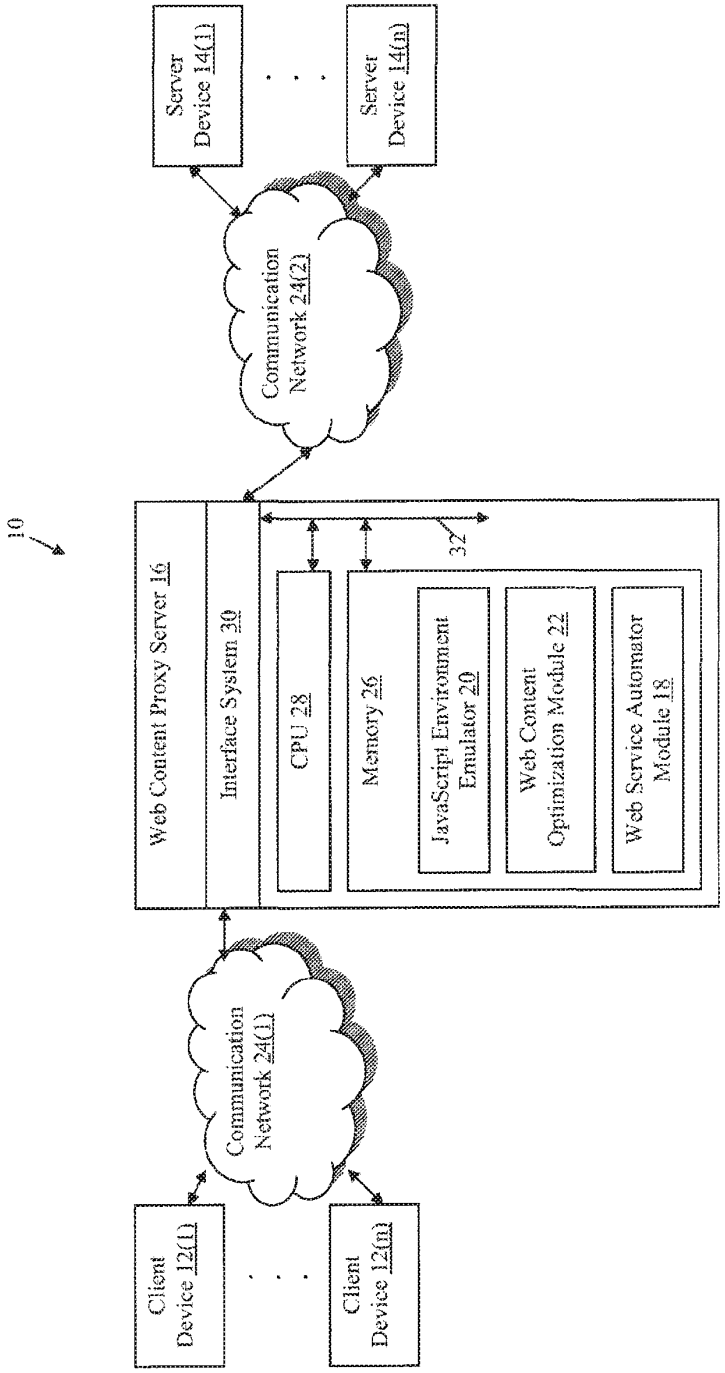
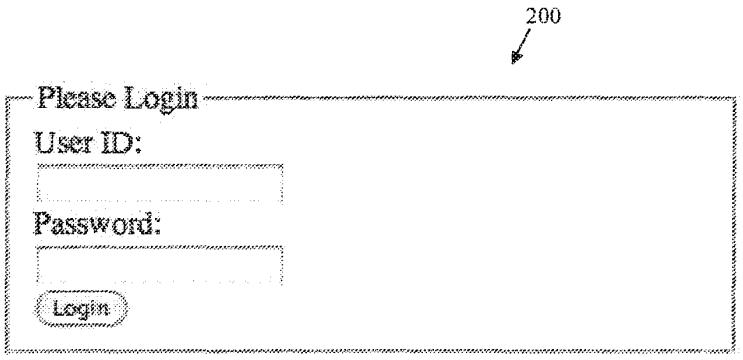
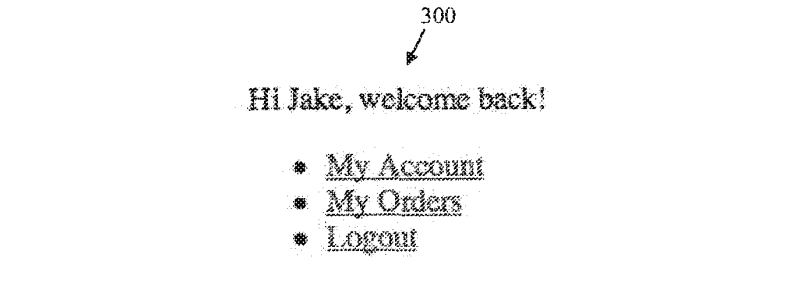

10
Server Device 14(1)
Server Device 14(n)
Communication Network 24(2)
Web Content Proxy Server 16
Interface System 30
CPU 28
Memory 26
32
JavaScript Environment Emulator 29
Web Content Optimization Module 22
Web Service Automator Module 18
Communication Network 24(1)
Client Device 12(1)
Client Device 12(n)

FIG. 1




```
<html>
    <head>
        <title>Login Page</title>
    </head>
    <body>
        <form id="login" method="post" action="/login.do">
            <fieldset>
                <legend>Please Login</legend>
                <input type="hidden" name="sessionid" value="dhfumsn3942b"/>
                <div>
                    User ID:<br/>
                    <input type="text" id="user" name="user"/>
                </div>
                <div>
                    Password:<br/>
                    <input type="text" id="passwd" name="passwd"/>
                </div>
                <input type="submit" value="Login"/>
            </fieldset>
        </form>
    </body>
</html>
```

FIG. 2



302

```
<html>
    <head>
        <title>You are now logged in</title>
    </head>
    <body>
        <div>Hi Jake, welcome back!</div>
        <ul>
            <li><a href="account.do">My Account</a></li>
            <li><a href="orders.do">My Orders</a></li>
            <li><a href="logout.do">Logout</a></li>
        </ul>

    </body>
</html>
```

FIG. 3

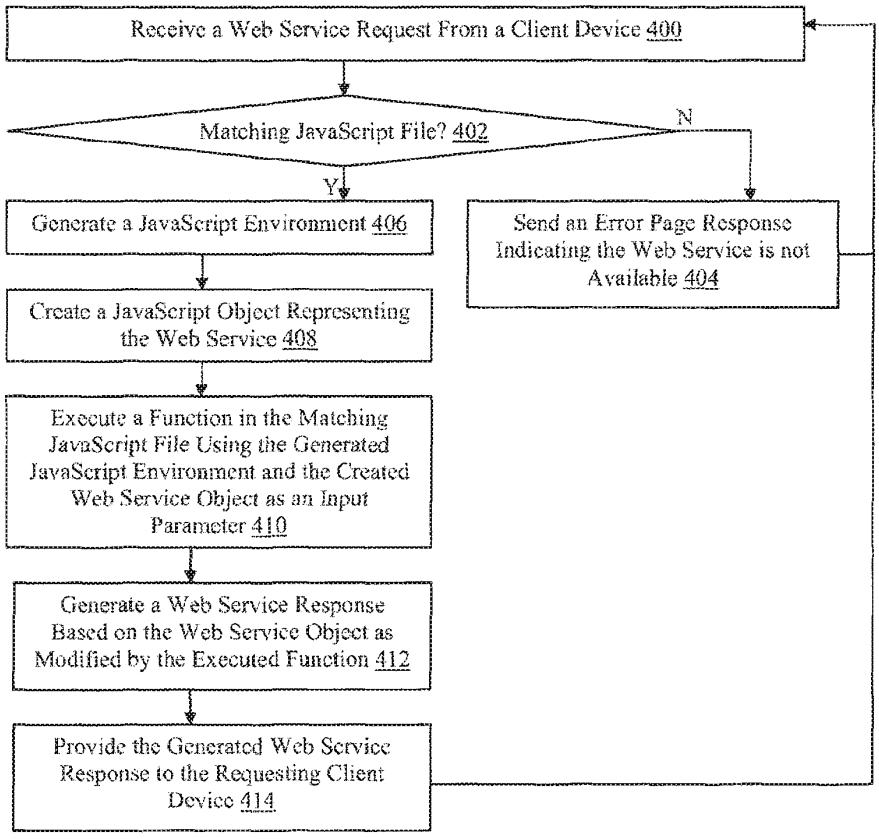

Receive a Web Service Request From a Client Device 400
Matching JavaScript File? 402
N
Y
Generate a JavaScript Environment 406
Send an Error Page Response Indicating the Web Service is not Available 404
Create a JavaScript Object Representing the Web Service 408
Execute a Function in the Matching JavaScript File Using the Generated JavaScript Environment and the Created Web Service Object as an Input Parameter 410
Generate a Web Service Response Based on the Web Service Object as Modified by the Executed Function 412
Provide the Generated Web Service Response to the Requesting Client Device 414

FIG. 4

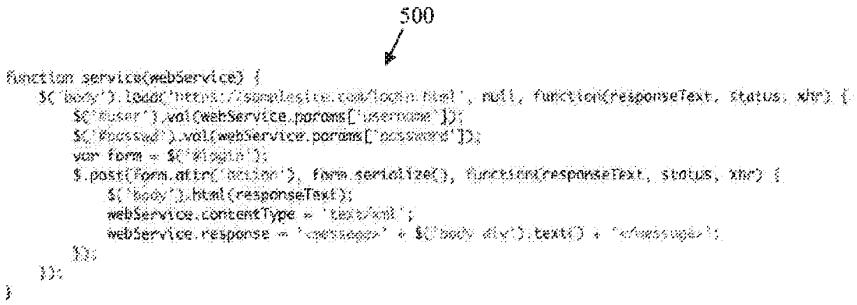

500
function service(webService) {
$('body').load('https://somesite.com/login.html', null, function(responseText, status, xhr) {
$('#user').val(webService.params['username']);
$('#passwd').val(webService.params['password']);
var form = $('#login');
$.post(form.attr('action'), form.serialize(), function(responseText, status, xhr) {
$('body').html(responseText);
webService.contentType = 'text/xml';
webService.response = '<message>' + $('body div').text() + '</message>';
});
});
}

FIG. 5

10
Server Device 14(1)
Server Device 14(n)
Communication Network 24(2)
Web Content Proxy Server 16
Interface System 30
CPU 28
Memory 26
32
JavaScript Environment Emulator 20
Web Content Optimization Module 22
Web Service Automator Module 18
Communication Network 24(1)
Client Device 12(1)
Client Device 12(n)

Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2843938 A1 2014/09/13

(21) 2 843 938

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(22) Date de dépôt/Filing Date: 2014/02/21

(41) Mise à la disp. pub./Open to Public Insp.: 2014/09/13

(30) Priorité/Priority: 2013/03/13 (US13/798,844)

(51) Cl.Int./Int.Cl. *H04L 29/06* (2006.01), *G06F 17/00* (2006.01), *H04L 12/16* (2006.01)

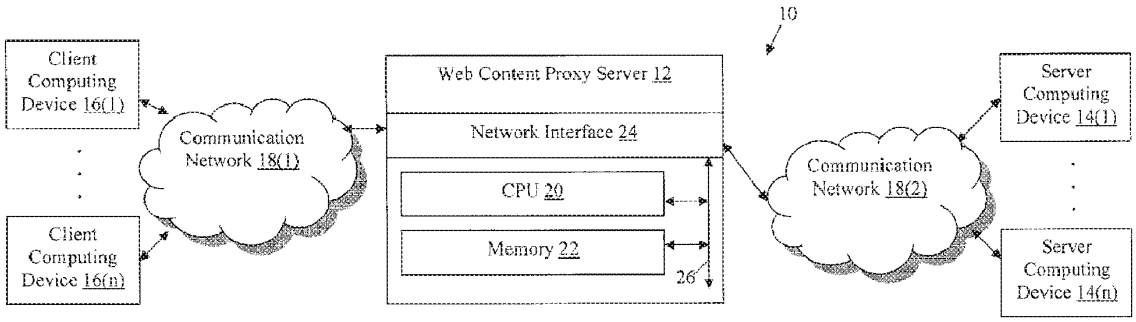
(71) Demandeur/Applicant:
USABLENET INC., US

(72) Inventeur/Inventor:
SCODA, ENRICO, IT

(74) Agent: PARLEE MCLAWS LLP

(54) Titre : PROCEDE DE TRAITEMENT DE FEUILLES DE STYLE EN CASCADE ET DISPOSITIFS DE CELUI-CI

(54) Title: METHODS FOR PROCESSING CASCADING STYLE SHEETS AND DEVICES THEREOF



(57) Abrégé/Abstract:
A method, non-transitory computer readable medium, and apparatus that processes cascading style sheets includes receiving a Hypertext Transfer Protocol (HTTP) request for a web page from a client computing device. A type of browser from which the HTTP request originated is identified based on a user agent header included in the HTTP request. At least one cascading style sheet (CSS) associated with the requested web page is obtained. The at least one CSS including a CSS property comprising at least a prefix followed by a name. A determination is made whether the prefix of the CSS property matches an established value. A transformation function for the CSS property is applied, when it is determined that the prefix of the CSS property matches the established value. The modified web page is sent to the requesting client computing device.

ABSTRACT

A method, non-transitory computer readable medium, and apparatus that processes cascading style sheets includes receiving a Hypertext Transfer Protocol (HTTP) request for a web page from a client computing device. A type of browser from which the HTTP request originated is identified based on a user agent header included in the HTTP request. At least one cascading style sheet (CSS) associated with the requested web page is obtained. The at least one CSS including a CSS property comprising at least a prefix followed by a name. A determination is made whether the prefix of the CSS property matches an established value. A transformation function for the CSS property is applied, when it is determined that the prefix of the CSS property matches the established value. The modified web page is sent to the requesting client computing device.

METHODS FOR PROCESSING CASCADING STYLE SHEETS AND DEVICES THEREOF

FIELD

[0001] This technology generally relates to methods, non-transitory computer readable medium, and apparatuses for processing cascading style sheets (CSSs) included in web pages and, more particularly, for reducing the amount of code required to define CSSs.

BACKGROUND

[0002] Cascading style sheets (CSSs) are used by software developers to define the format, appearance, and/or layout of web pages or to define media associated with web pages, such as animations. The interpretation of the properties of CSS code is browser-specific. For compatibility, developers often include multiples of each property and the rendering web browser utilizes only the properties it recognizes. An exemplary CSS code fragment 300 is illustrated in FIG. 3 as including six CSS properties 302(1)-302(6). The CSS properties 302(1)-302(4) are used by WebKit-based web browsers, the CSS property 302(5) is used by Mozilla Firefox™ web browsers, and the CSS property 302(6) is used by standards-compliant web browsers to define the "rounded_border" class attribute.

[0003] Due to the multiple properties, defining CSSs can require a large amount of code, which may be complex and/or difficult to maintain. Communicating and rendering web pages with CSSs can also require a relatively large amount of resources and time. Additionally, the implementations required for compatibility are exposed to the client computing devices, which may not be desirable.

SUMMARY

[0004] A method for processing cascading style sheets includes receiving at a web content proxy server a Hypertext Transfer Protocol (HTTP) request for a web

page from a client computing device. A type of browser from which the HTTP request originated is identified with the web content proxy server based on a user agent header included in the HTTP request. At least one cascading style sheet (CSS) associated with the requested web page is obtained with the web content proxy server. The at least one CSS including a CSS property comprising at least a prefix followed by a name. A determination is made whether the prefix of the CSS property matches an established value with the web content proxy server. A transformation function for the CSS property is applied with the web content proxy server, when it is determined that the prefix of the CSS property matches the established value. The transformation function is configured to modify the web page to replace the CSS property based on the identified type of browser and the name of the CSS property. The modified web page is sent with the web content proxy server to the requesting client computing device.

[0005] A non-transitory computer readable medium having stored thereon instructions for processing cascading style sheets including machine executable code which when executed by a processor, causes the processor to perform steps including receiving an HTTP request for a web page from a client computing device. A type of browser from which the HTTP request originated is identified based on a user agent header included in the HTTP request. At least one cascading style sheet (CSS) associated with the requested web page is obtained. The at least one CSS including a CSS property comprising at least a prefix followed by a name. A determination is made whether the prefix of the CSS property matches an established value. A transformation function for the CSS property is applied, when it is determined that the prefix of the CSS property matches the established value. The transformation function is configured to modify the web page to replace the CSS property based on the identified type of browser and the name of the CSS property. The modified web page is sent to the requesting client computing device.

[0006] A web content proxy server apparatus includes a memory coupled to a processor which is configured to execute programmed instructions stored in the

memory comprising receiving as HTTP request for a web page from a client computing device. A type of browser from which the HTTP request originated is identified based on a user agent header included in the HTTP request. At least one cascading style sheet (CSS) associated with the requested web page is obtained. The at least one CSS including a CSS property comprising at least a prefix followed by a name. A determination is made whether the prefix of the CSS property matches an established value. A transformation function for the CSS property is applied, when it is determined that the prefix of the CSS property matches the established value. The transformation function is configured to modify the web page to replace the CSS property based on the identified type of browser and the name of the CSS property. The modified web page is sent to the requesting client computing device.

[0007] This technology provides a number of advantages including methods, non-transitory computer readable medium, and apparatuses that reduce the amount and complexity of code required to define web page CSSs. With this technology, developers can include a prefix to indicate that a CSS property should be transformed server-side to be compatible with a web browser of a client computing device that originated the request for the web page with the CSS. Accordingly, CSS code can be optimized and web pages with CSSs can be communicated relatively quickly to requesting client computing devices.

BRIEF DESCRIPTION OF THE DRAWINGS

[0008] FIG. 1 is a block diagram of a network environment which incorporates an exemplary web content proxy server;

[0009] FIG. 2 is a flowchart of an exemplary method for processing a cascading style sheet (CSS) included in a web page;

[0010] FIG. 3 is an exemplary cascading style sheet (CSS) code fragment defining a class attribute for a plurality of web browsers;

[0011] FIG. 4 is an exemplary CSS code fragment defining the class attribute of FIG. 3 and including a CSS property with an exemplary prefix indicating that the property requires transformation;

[0012] FIG. 5 is an exemplary CSS code fragment defining the class attribute of FIG. 3 and including the CSS property of FIG. 4 transformed for a standards-compliant web browser;

[0013] FIG. 6 is an exemplary CSS code fragment defining the class attribute of FIG. 3 and including the CSS property of FIG. 4 transformed for a Mozilla Firefox™ web browser; and

[0014] FIG. 7 is an exemplary CSS code fragment defining the class attribute of FIG. 3 and including the CSS property of FIG. 4 transformed for a WebKit-based web browser.

DETAILED DESCRIPTION

[0015] An exemplary network environment 10 is illustrated in FIG. 1 as including an exemplary web content proxy server 12. In this example, the web content proxy server 12 is coupled to a plurality of server computing devices 14(1)-14(n) and a plurality of client computing devices 16(1)-16(n) by communication networks 18(1) and 18(2), although other types and numbers of devices, components, and elements in other topologies could be used. This technology provides a number of advantages including methods, non-transitory computer readable medium, and apparatuses for more efficiently processing cascading style sheets (CSSs) of a web page based on the web browser originating the request for the web page and without requiring that the web page include multiple properties of the CSS for compatibility.

[0016] Referring more specifically to FIG. 1, the web content proxy server 12 includes at least one processor or central processing unit (CPU) 20, a memory 22, and a network interface 24, which are coupled together by a bus 26 or other link, although other numbers and types of components, parts, devices, systems, and elements in

other configurations and locations can also be used. The processor 20 in the web content proxy server 12 may execute a program of stored instructions one or more aspects of the present invention, as described and illustrated by way of the embodiments herein, although the processor 20 could execute other numbers and types of programmed instructions.

[0017] The memory 22 in the web content proxy server 12 stores these programmed instructions for one or more aspects of the present invention as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a RAM or a ROM in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other non-transitory computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor 20, can be used for the memory 22 in the web content proxy server 12.

[0018] The network interface 24 in the web content proxy server 12 is used to operatively couple and communicate between the web content proxy server 12 and the client devices 16(1)-16(n) and server devices 14(1)-14(n) via the communication networks 24(1) and 24(2), although other types and numbers of networks with other types and numbers of connections and configurations can also be used. For example, one or more of the communication networks 24(1) and 24(2) can include one or more networks, such as one or more local area networks (LANs) and/or wide area networks (WANs) such as the Internet. By way of example only, the communication networks can use TCP/IP over Ethernet and industry-standard protocols, including Hypertext transfer protocol (HTTP), secure HTTP (HTTPS), wireless application protocol (WAP), and/or SOAP, although other types and numbers of communication networks having their own communications protocols, can also be used.

[0019] Generally, the server computing devices 14(1)-14(n) process requests for web pages and other web content received from the client computing devices 16(1)-16(n) via the communication networks 18(1) and 18(2) according to the HTTP-

based protocol, for example, although the server computing devices 14(1)-14(n) can also provide other numbers and types of functions. Each of the server computing devices 14(1)-14(n) can include a at least one processor or CPU, a memory, and a network interface, which are coupled together by a bus or other link, although each of the server computing devices 14(1)-14(n) could have other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor in each of the server computing devices 14(1)-14(n) executes a program of stored instructions for one or more aspects of the present invention, as described and illustrated by way of the embodiments herein, although the processor could execute other numbers and types of programmed instructions.

[0020] The memory in each of the server computing devices 14(1)-14(n) stores these programmed instructions for one or more aspects of the present invention, as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a RAM or a ROM in the system or a floppy disk, hard disk, CD ROM, DVD ROM, or other non-transitory computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to the processor, can be used for the memory in each of the server computing devices 14(1)-14(n).

[0021] The network interface in each of the server computing devices 14(1)-14(n) is used to operatively couple and communicate between the server computing devices 14(1)-14(n) and the web content proxy server 12 and client computing devices 16(1)-16(n) via the communication networks 24(1) and 24(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used. Each of the server computing devices 16(1)-16(n) may be hardware or software or may represent a system with multiple servers in a pool, which may include internal or external networks. In this example the each of the server computing devices 16(1)-16(n) may be any version of

Microsoft® IIS server or Apache® server, although other types of servers may be used.

[0022] The client computing devices 12(1)-12(n) enable a user to request, receive, and interact with applications, web services, and content hosted by the server computing devices 14(1)-14(n) through the web content proxy server 12 via the communication networks 24(1) and 24(2), although one or more of the client computing devices 12(1)-12(n) could access content and utilize other types and numbers of applications from other sources and could provide a wide variety of other functions for the user. By way of example only, the client computing devices 12(1)-12(n) can be smart phones, personal digital assistants, or computers.

[0023] In this example, each of the client computing devices 16(1)-16(n) includes at least one processor or a CPU, a memory, a network interface, a user input device, and a display, which are coupled together by a bus or other link, although one or more of client computing devices 16(1)-16(n) can include other numbers and types of components, parts, devices, systems, and elements in other configurations. The processor in each of the client computing devices 16(1)-16(n) can execute a program of instructions stored in the memory of each of the client computing devices 16(1)-16(n) for one or more aspects of the present invention as described and illustrated herein, although the processor could execute other numbers and types of programmed instructions.

[0024] The memory in each of the client computing devices 16(1)-16(n) stores these programmed instructions for one or more aspects of the present invention, as described and illustrated herein, although some or all of the programmed instructions could be stored and/or executed elsewhere. A variety of different types of memory storage devices, such as a RAM or a ROM in the system or a floppy disk, hard disk, CD ROM, or other non-transitory computer readable medium which is read from and/or written to by a magnetic, optical, or other reading and/or writing system that is coupled to processor can be used for the memory in each of the client computing devices 16(1)-16(n). Each of the client computing devices 16(1)-16(n)

can be configured to access web services and content through a web browser stored in the memory.

[0025] The network interface in each of the client computing devices 16(1)-16(n) is used to operatively couple and communicate between each of the client computing devices 16(1)-16(n) and the web content proxy server 12 and server computing devices 12(1)-12(n) via the communication networks 18(1) and 18(2), although other types and numbers of communication networks with other types and numbers of connections and configurations can be used.

[0026] The user input device in each of the client computing devices 16(1)-16(n) can be used to input selections, such as a request for a particular web page, although the user input device could be used to input other types of requests and data and interact with other elements. The user input device in each of the client computing devices 16(1)-16(n) can include a keypad, touch screen, and/or vocal input processing system, although other types and numbers of user input devices can also be used.

[0027] The display in each of the client computing devices 16(1)-16(n) can be used to show data and information to the user, such as a requested web page by way of example only. The display in each of the client computing devices 16(1)-16(n) can be an LCD, LED, or OLED display, for example, although other types and numbers of displays could be used depending on the particular type of client computing device 16(1)-16(n).

[0028] Although embodiments of the web content proxy server 12, client computing devices 16(1)-16(n), and server computing devices 14(1)-14(n) are described and illustrated herein, each of the web content proxy server 12, client computing devices 16(1)-16(n), and server computing devices 14(1)-14(n) can be implemented on any suitable computer apparatus or computing device. It is to be understood that the apparatuses and devices of the embodiments described herein are for exemplary purposes, as many variations of the specific hardware and software

used to implement the embodiments are possible, as will be appreciated by those skilled in the relevant art(s). Furthermore, each of the devices of the embodiments may be conveniently implemented using one or more general purpose computers, microprocessors, digital signal processors, and micro-controllers, programmed according to the teachings of the embodiments, as described and illustrated herein, and as will be appreciated by those ordinary skill in the art.

[0029] In addition, two or more computing apparatuses or devices can be substituted for any one of the devices in any embodiment described herein. Accordingly, principles and advantages of distributed processing, such as redundancy and replication also can be implemented, as desired, to increase the robustness and performance of the devices of the embodiments. The embodiments may also be implemented on computer apparatuses or devices that extend across any suitable network using any suitable interface mechanisms and communications technologies, including by way of example only telecommunications in any suitable form (e.g., voice and modem), wireless communications media, wireless communications networks, cellular communications networks, G3 communications networks, Public Switched Telephone Network (PSTNs), Packet Data Networks (PDNs), the Internet, intranets, and combinations thereof.

[0030] The examples may also be embodied as a non-transitory computer readable medium having instructions stored thereon for one or more aspects of the present technology as described and illustrated by way of the examples herein, as described herein, which when executed by a processor, cause the processor to carry out the steps necessary to implement the methods of the examples, as described and illustrated herein.

[0031] An exemplary method for processing cascading style sheets (CSSs) included in a web page will now be described with reference to FIGS. 1-7. Referring specifically to FIG. 2, in step 200, the web content proxy server 12 receives a Hypertext Transfer Protocol (HTTP) request for a web page from one of the client computing devices 16(1)-16(n) via the communication network 18(1).

[0032] In step 202, the web content proxy server 12 identifies a type of web browser from which the HTTP request received in step 200 originated. In this example, the HTTP request can include a user agent header which includes information regarding the originating web browser including at least the type of the web browser. In step 204, the web content proxy server 12 retrieves the requested web page from one of the server computing devices 14(1)-14(n), although the web page could be retrieved from other types of computing devices.

[0033] In step 206, the web content proxy server 12 determines whether the retrieved web page has an associated CSS. In this example the CSS is included in the source code for the web page. However, in other examples, the web page source code references a separate external file which includes a CSS. In examples in which a CSS is defined in a separate file, the initially-requested web page source code may cause a web browser of the requesting one of the client computing device 16(1)-16(n) to submit a request for the file. The web content proxy server 12 can parse the web page or the file retrieved from the one of the server computing devices 14(1)-14(n) to identify character strings associated with a CSS included therein, for example, although other methods of determining whether the web page has an associated CSS can also be used.

[0034] If the web content proxy server 12 determines that the retrieved web page does not have an associated CSS, then the No branch is taken to step 208. In step 208, the retrieved web page is sent to the requesting one of the client computing devices 16(1)-16(n). Optionally, the web content proxy server 12 can perform other optimizations on the web page prior to forwarding the web page to the requesting one of the client computing devices 16(1)-16(n), such as optimizing the web page for display on a mobile computing device, for example.

[0035] Referring back to step 206, if the web content proxy server 12 determines that the retrieved web page does have an associated CSS, then the Yes branch is taken to step 210. In step 210, the web content proxy server 12 determines whether a prefix of one of the CSS properties of the CSS matches an established

value. The established value can be set by an administrator and stored in the memory 22 of the web content proxy server 12, for example. The established value can also be shared with web page developers. If the web content proxy server 12 determines that the prefix of the CSS property does match the established value, then the Yes branch is taken to step 212.

[0036] Referring to FIG. 4, an exemplary CSS code fragment 400 defining the "rounded_border" class attribute of FIG. 3 is illustrated. In this example, the CSS code fragment 400 includes a CSS property 402 with a "-u-" prefix. In this example, the "-u-" prefix matches the established value, although the established value can be any value that does not match a prefix used by a web browser (e.g. "-webkit-" or "-moz-").

[0037] Referring back to FIG. 2, in step 212, the web content proxy server 12 determines whether a transformation function is stored in the memory 22 for the CSS property based on the type of web browser identified in step 202 and a name of the CSS property (e.g. "border-radius" of the CSS property 402). In this example, the web content proxy server 12 can store transformation functions for a plurality of CSS properties and a plurality of web browsers. The transformation functions are configured to modify the web page to replace a CSS property with one or more CSS properties that are recognized by the web browser of the requesting one of the client computing devices 16(1)-16(n) and that provide equivalent functionality. If the web content proxy server 12 determines that there is a transformation function for the CSS property and the identified type of web browser, then the Yes branch is taken to step 214.

[0038] In step 214, the web content proxy server 12 applies the transformation function to the CSS property of the web page. Referring to FIGS. 5-7, exemplary CSS code fragments 500, 600, and 700 defining the "rounded_border" class attribute of FIG. 3 are illustrated. The CSS code fragments 500, 600, and 700 include the CSS property 402 of FIG. 4 transformed for a standards-compliant, Mozilla Firefox™, and WebKit-based type of web browser, respectively, into the CSS properties 502, 602,

and 702. Although one example of a transformation and web browser is illustrated and described here, other types of transformations and other types web browsers can also be used.

[0039] In the examples shown in FIGS. 5-7, the applied transformations replace the prefix of the CSS property, but in another example one or more of the CSS property prefix, name, or value (e.g. "5px 10px 15px 20px" of the CSS property 402) can be removed, replace, or modified. Additionally, the CSS property 402 is an example of a CSS rule, but the CSS property can also be a CSS extension function or any other type of CSS statement or declaration.

[0040] In step 216, the web content proxy server 12 determines whether there is an additional CSS property included in the CSS. For example, the CSS can include multiple class attributes having at least one CSS property and/or one or more class attributes with multiple CSS properties. If the web content proxy server 12 determines that the CSS includes an additional CSS property, then the Yes branch is taken to step 210. Optionally, in this example steps 210-214 would be repeated for each CSS property included in the CSS.

[0041] If in step 216 the web content proxy server 12 determines that there is no additional CSS property, then the No branch is taken to step 218. In step 218, the web content proxy server 12 determines whether the retrieved web page includes an additional CSS. If the web content proxy server 12 determines that there is an additional CSS, then the Yes branch is taken to step 210. Optionally, in this example steps 210-216 would again be repeated for each additional CSS included in the retrieved web page.

[0042] If, in step 218, the web content proxy server 12 determines there is no additional CSS, then the No branch is taken to step 208. In step 208, the web page, which may have come from step 206 or may have been modified based on any transformation functions applied in step 214, is sent to the requesting one of the client computing devices 16(1)-16(n).

[0043] Referring back to step 210, if the web content proxy server 12 determines that the prefix of the CSS property does not match the established value, then the No branch is taken to step 216. In step 216, the web content proxy server 12 determines whether the CSS includes an additional CSS property, as described and illustrated earlier.

[0044] Additionally, referring back to step 212, if the web content proxy server 12 determines that there is not a transformation function for the CSS property and the identified type of web browser, then the No branch is optionally taken to step 220. In step 220, the web content proxy server 12 removes the CSS property from the retrieved web page. Since the CSS property includes a prefix matching an established value and that the web browser of the requesting one of the client computing devices 16(1)-16(n) will not recognize, the CSS property will not be interpreted by the web browser and can be removed. Once removed, the web content proxy server 12 determines, in step 216, whether the CSS includes an additional CSS property, as described and illustrated earlier.

[0045] Accordingly, as illustrated and described herein this technology provides a number of advantages including improved methods, non-transitory computer readable medium, and apparatuses for reducing the amount of code required to define web page CSSs. With this technology, developers can indicate, using an established prefix value, CSS properties that are to be transformed to be compliant with a web browser from which a request for the web page included the CSS originated. Thereby, developers can define web page CSSs using less and simpler code, which can advantageously be optimized server-side for each requesting web browser.

[0046] Having thus described the basic concept of the invention, it will be rather apparent to those skilled in the art that the foregoing detailed disclosure is intended to be presented by way of example only, and is not limiting. Various alterations, improvements, and modifications will occur and are intended to those skilled in the art, though not expressly stated herein. These alterations,

improvements, and modifications are intended to be suggested hereby, and are within the spirit and scope of the invention. Additionally, the recited order of processing elements or sequences, or the use of numbers, letters, or other designations therefore, is not intended to limit the claimed processes to any order except as may be specified in the claims. Accordingly, the invention is limited only by the following claims and equivalents thereto.

CLAIMS

What is claimed is:

1. A method for processing cascading style sheets, comprising:

receiving at a web content proxy server a Hypertext Transfer Protocol (HTTP) request for a web page from a client computing device;

identifying with the web content proxy server a type of browser from which the HTTP request originated based on a user agent header included in the HTTP request;

obtaining with the web content proxy server at least one cascading style sheet (CSS) associated with the requested web page, the at least one CSS including a CSS property, the CSS property comprising at least a prefix followed by a name;

determining with the web content proxy server whether the prefix of the CSS property matches an established value;

applying with the web content proxy server a transformation function for the CSS property, the transformation function configured to modify the CSS to replace the CSS property based on the identified type of browser and the name of the CSS property, when the determining indicates a match between the prefix of the CSS property and the established value; and

sending with the web content proxy server the modified CSS to the requesting client computing device.

2. The method as set forth in claim 1, further comprising:

determining with the web content proxy server whether there is a transformation function for the CSS property based on the identified type of browser and the name of the CSS property, when it is determined that the prefix of the CSS property matches the established value; and

removing with the web content proxy server the CSS property, when the determining indicates a match between the prefix of the CSS property and the established value.

3. The method as set forth in claim 1, wherein the CSS property further comprises at least one value following the name of the CSS property and the transformation function is further configured to replace the prefix or the value of the CSS property.

4. The method as set forth in claim 1, further comprising:

determining with the web content proxy server whether the at least one CSS includes one or more other CSS properties; and

repeating with the web content proxy server the determining whether the prefix of the one or more other CSS properties matches the established value and the applying the transformation function for the CSS property for each of the one or more other CSS properties prior to sending the modified CSS to the requesting client computing device.

5. The method as set forth in claim 1, wherein the CSS property is a CSS rule or a CSS extension function.

6. A non-transitory computer readable medium having stored thereon instructions for processing cascading style sheets comprising machine executable code which when executed by a processor, causes the processor to perform steps comprising:

receiving a Hypertext Transfer Protocol (HTTP) request for a web page from a client computing device;

identifying a type of browser from which the HTTP request originated based on a user agent header included in the HTTP request;

obtaining at least one cascading style sheet (CSS) associated with the requested web page, the at least one CSS including a CSS property, the CSS property comprising at least a prefix followed by a name;

determining whether the prefix of the CSS property matches an established value;

applying a transformation function for the CSS property, the transformation function configured to modify the CSS to replace the CSS property based on the identified type of browser and the name of the CSS property, when the determining indicates a match between the prefix of the CSS property and the established value; and

sending the modified CSS to the requesting client computing device.

7. The medium as set forth in claim 6, further having stored thereon instructions comprising machine executable code which when executed by the processor causes the processor to perform steps further comprising:

determining whether there is a transformation function for the CSS property based on the identified type of browser and the name of the CSS property, when the determining indicates a match between the prefix of the CSS property and the established value; and

removing the CSS property, when it is determined that there is not a transformation function for the CSS property.

8. The medium as set forth in claim 6, wherein the CSS property further comprises at least one value following the name of the CSS property and the transformation function is further configured to replace the prefix or the value of the CSS property.

9. The medium as set forth in claim 6, further having stored thereon instructions comprising machine executable code which when executed by the processor causes the processor to perform steps further comprising:

determining whether the at least one CSS includes one or more other CSS properties; and

repeating the determining whether the prefix of the one or more other CSS properties matches the established value and the applying the transformation function for the CSS property for each of the one or more other CSS properties prior to sending the modified CSS to the requesting client computing device.

10. The medium as set forth in claim 6, wherein the CSS property is a CSS rule or a CSS extension function.

11. A web content proxy server apparatus, comprising:

a memory coupled to a processor which is configured to execute programmed instructions stored in the memory comprising:

receiving a Hypertext Transfer Protocol (HTTP) request for a web page from a client computing device;

identifying a type of browser from which the HTTP request originated based on a user agent header included in the HTTP request;

obtaining at least one cascading style sheet (CSS) associated with the requested web page, the at least one CSS including a CSS property, the CSS property comprising at least a prefix followed by a name;

determining whether the prefix of the CSS property matches an established value;

applying a transformation function for the CSS property, the transformation function configured to modify the CSS to replace the CSS property based on the identified type of browser and the name of the CSS