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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
30 January 2014 (30.01.2014)

WIPO | PCT

(10) International Publication Number
WO 2014/018780 A1

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

(21) International Application Number:
PCT/US2013/052106

(22) International Filing Date:
25 July 2013 (25.07.2013)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/675,492 25 July 2012 (25.07.2012) US

(71) Applicant: **INDIX CORPORATION** [US/US]; 17723 SE 58th Pl, Bellevue, WA 98006 (US).

(72) Inventors: **KALIKIVAYI, Satyanarayana Rao**; 17723 SE 58th Pl, Bellevue, WA 98006 (US). **MUPPALLA, Rajesh**; 17723 SE 58th Pl, Bellevue, WA 98006 (US). **PARTHASARATHY, Sanjay**; 17723 SE 58th Pl, Bellevue, WA 98006 (US).

(74) Agent: **PHILIPP, Adam L.K.**; Aeon Law, 1218 3rd Avenue, Suite 2100, Seattle, WA 98101 (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, IS, JP, KE, KG, 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, PA, 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.

[Continued on next page]

(54) Title: ADAPTIVE GATHERING OF STRUCTURED AND UNSTRUCTURED DATA SYSTEM AND METHOD

(57) **Abstract**: Content is obtained from a webpage accessed via a URI, which URI is obtained from a URI queue. The content is parsed for price and product information according to a parse map, with the resulting parse result being stored. The priority of URIs in the URI queue is adjusted based on analysis of the parse result for changes in price and product attributes and according to other criteria. The parse map may be one associated with the URI or a general purpose parse maps. The parse result may be validated by human- and machine-based systems, including by graphically labeling price and product information in the content for human confirmation or correction.

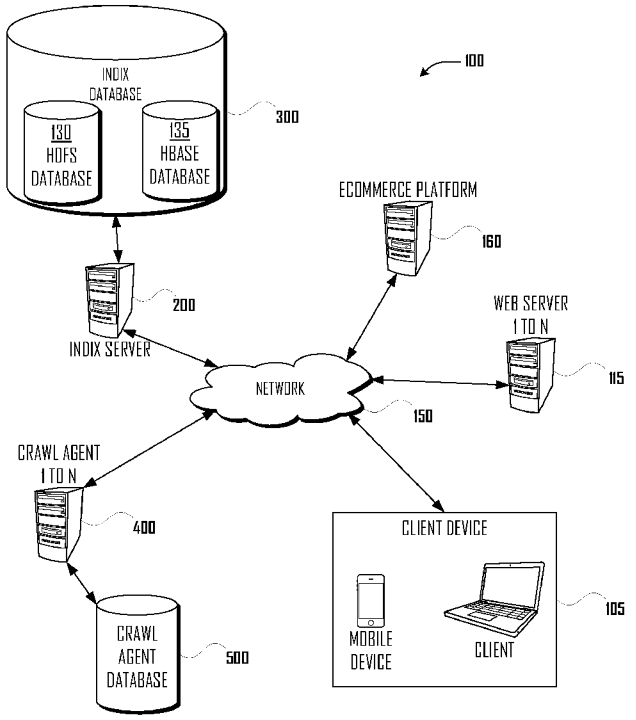

Fig.1

WO 2014/018780 A1

(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, KM, 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))*

TITLE

Adaptive Gathering of Structured and Unstructured Data System and Method

FIELD

[Para 01] This disclosure relates to a method and system to efficiently obtain information from third-party sources on the Internet and to parse the information into price, product, and other information.

BACKGROUND

[Para 02] The following description includes information that may be useful in understanding the present invention. It is not an admission that any of the information provided herein is prior art or relevant to the presently claimed invention, or that any publication specifically or implicitly referenced is prior art.

[Para 03] Search engines, such as Google, Bing, and others search and index vast quantities of information on the Internet. "Crawlers" (a.k.a. "spiders") follow URLs obtained from a "queue" to obtain content, usually from web pages. The crawlers or other software store and index some of the content. Users can then search the indexed content, view results, and follow hyperlinks back to the original source or to the stored content (the stored content often being referred to as a "cache"). Computing resources to crawl and index, however, are not limitless. The URL queues are commonly prioritized to direct crawler resources to web page servers which can accommodate the traffic, which do not block crawlers (such as according to "robots.txt" files commonly available from webpage servers), which experience greater traffic from users, and which experience more change in content.

[Para 04] Conventional search engines, however, are not focused on price and product information. If a price changes on a webpage, but the rest of the webpage remains the same, traditional crawlers (or the queue manager) will not prioritize the webpage position in the queue, generally because the price is a tiny fraction of the overall content and the change is not labeled as being significant; conversely, if the webpage changes, but the price and/or product information remains the same, the change in webpage content may cause a traditional crawler to prioritize the webpage position in the queue due to the overall change in content, notwithstanding that that price and product information remained the same.

BRIEF DESCRIPTION OF THE DRAWINGS

[Para 05] Figure 1 is a network and device diagram illustrating exemplary computing devices configured according to embodiments disclosed in this paper.

[Para 06] Figure 2 is a functional block diagram of an exemplary Indix Server 200 computing device and some data structures and/or components thereof.

[Para 07] Figure 3 is a functional block diagram of the Indix Datastore 300 illustrated in the computing device of Figure 2.

[Para 08] Figure 4 is a functional block diagram of an exemplary Crawl Agent 400 computing device and some data structures and/or components thereof.

[Para 09] Figure 5 is a functional block diagram of the Crawl Agent Datastore 500 illustrated in the computing device of Figure 4.

[Para 10] Figure 6 is a flowchart illustrating an embodiment of a URI Check Routine 600 in which the Crawl Agent 400 obtains a URI 305 from a URI Queue 355 and obtains a URI-Content Instance 310.

[Para 11] Figure 7 is a flowchart illustrating an embodiment of a Parser Routine 700 for parsing a URI-Content Instance 310 and saving a Parse Result 325.

[Para 12] Figure 8 is a flowchart illustrating an embodiment of a Seeder Routine 800 for identifying URIs 305 which contain Price or Product Attributes and adding the URIs to the URI Queue 355.

[Para 13] Figures 9A and 9B are flowcharts illustrating an embodiment of a URI Queue Manager Routine 900.

[Para 14] Figure 10 is a flowchart illustrating an embodiment of a Parse Map Validation Routine 1000.

[Para 15] Figure 11 is an illustration of a browser window showing a webpage with HTML and CSS elements corresponding to Attributes being labeled with Attribute names.

[Para 16] Figure 12 is a flowchart illustrating an embodiment of an MPID Assigner Routine 1200.

DETAILED DESCRIPTION

[Para 17] The following Detailed Description provides specific details for an understanding of various examples of the technology. One skilled in the art will understand that the technology may be practiced without many of these details. In some instances, structures and functions have not been shown or described in detail or at all to avoid unnecessarily obscuring the description of the examples of the technology. It is intended that the terminology used in the description presented below be interpreted in its broadest reasonable manner, even though it is being used in conjunction with a detailed description of certain examples of the technology. Although certain terms may be emphasized below, any terminology intended to be interpreted in any restricted manner will be overtly and specifically defined as such in this Detailed Description section.

[Para 18] Unless the context clearly requires otherwise, throughout the description and the claims, the words "comprise," "comprising," and the like are to be construed in an inclusive sense, as opposed to an exclusive or exhaustive sense; that is to say, in the sense of "including, but not limited to." As used herein, the term "connected," "coupled," or any variant thereof means any connection or coupling, either direct or indirect between two or more elements; the coupling of connection between the elements can be physical, logical, or a combination thereof. Additionally, the words, "herein," "above," "below," and words of similar import, when used in this application, shall refer to this application as a whole and not to particular portions of this application. When the context permits, words using the singular may also include the plural while words using the plural may also include the singular. The word "or," in reference to a list of two or more items, covers all of the following interpretations of the word: any of the items in the list, all of the items in the list, and any combination of one or more of the items in the list.

[Para 19] Certain elements appear in several of the Figures with the same capitalized element text, but a different element number. When referred to herein with the capitalized element text but with no element number, these references should be understood to be largely equivalent and to refer to any of the elements with the same capitalized element text, though potentially with differences based on the computing device within which the various embodiments of the element appears.

[Para 20] As used herein, a Uniform Resource Identifier ("URI") is a string of characters used to identify a resource on a computing device and/or a network, such as the Internet. Such identification enables interaction with representations of the resource using specific protocols. “Schemes” specifying a syntax and associated protocols define each URI.

[Para 21] The generic syntax for URI schemes is defined in Request for Comments (“RFC”) memorandum 3986 published by the Internet Engineering Task Force (“IETF”). According to RFC 3986, a URI (including a URL) consists of four parts:

<scheme name>: <hierarchical part> [? <query>] [# <fragment>]

[Para 22] A URI begins with a scheme name that refers to a specification for assigning identifiers within that scheme. The scheme name consists of a letter followed by any combination of letters, digits, and the plus (“+”), period (“.”), or hyphen (“-”) characters; and is terminated by a colon (“:”).

[Para 23] The hierarchical portion of the URI is intended to hold identification information that is hierarchical in nature. Often this part is delineated with a double forward slash (“//”), followed by an optional authority part and an optional path.

[Para 24] The optional authority part holds an optional user information part (not shown) terminated with “@” (e.g. username:password@), a hostname (i.e., domain name or IP address, here “example.com”), and an optional port number preceded by a colon “:”.

[Para 25] The path part is a sequence of one or more segments (conceptually similar to directories, though not necessarily representing them) separated by a forward slash (“/”). If a URI includes an authority part, then the path part may be empty.

[Para 26] The optional query portion is delineated with a question mark and contains additional identification information that is not necessarily hierarchical in nature. Together, the path part and the query portion identify a resource within the scope of the URI’s scheme and authority. The query string syntax is not generically defined, but is commonly organized as a sequence of zero or more <key>=<value> pairs separated by a semicolon or ampersand, for example:

key1=value1;key2=value2;key3=value3 (Semicolon), or

key1=value1&key2=value2&key3=value3 (Ampersand)

[Para 27] Much of the above information is taken from RFC 3986, which provides additional information related to the syntax and structure of URIs. RFC 3986 is hereby incorporated by reference, for all purposes.

[Para 28] As used herein, a “Crawl Agent” is a process, generally executed on or by a server, which requests content from other servers on the Internet, often though not exclusively relative to the World Wide Web. An example of Crawl Agents 1 to N is illustrated in Figure 1 as Crawl Agent 400; multiple Crawl Agents 400 may be present and may be represented by this illustration.

[Para 29] As used herein, "Product" shall be understood to mean "products or services." References to "Product Attribute" herein shall be understood to mean "product or service attribute."

[Para 30] As used herein, an "iPID" or iPID 330 is a unique identifier assigned within the Indix System to a URI for a product. The iPID 330 may be, for example, a hash of URI 305.

[Para 31] As used herein, a "Master iPID" or "MPID" or MPID 332 is an iPID 330 assigned to a Product by the MPID Assigner Routine 1200. An MPID is generally meant to identify a single Product, generally produced by a common manufacturer, though the Product may be distributed and sold by multiple parties.

[Para 32] iPIDs and MPIDs are associated with Price Attribute 340 records and Product Attribute 345 records.

[Para 33] A Price Attribute 340 record may comprise one or more records comprising, for example, values which encode an iPRID which may be an identifier for a price observed at a particular time, an iPID (discussed above), a Product Name (a "Product Name" value in this record may also be referred to herein as a "Product"), a Standard Price, a Sale, a Price, a Rebate amount, a Price Instructions record (containing special instructions relating to a price, such as that the price only applies to students), a Currency Type, a Date and Time Stamp, a Tax record, a Shipping record (indicating costs relating to shipping to different locations, whether tax is calculated on shipping costs, etc.), a Price Validity Start Date, a Price Validity End Date, a Quantity, a Unit of Measure Type, a Unit of Measure Value, a Merchant Name (with the name of a merchant from whom the Product is available; a "Merchant Name" value in this record may also be referred to herein as a "Merchant"), a Store Name (a Merchant may have multiple stores; a "Store Name" value in this record may also be referred to herein as a "Store"), a User ID, a Data Channel (indicating the source of the Price Attribute 340 record, such as an online crawl, a crowdsource, a licensed supplier of price information, or from a merchant), a Source Details record (for example, indicating a URI, a newspaper advertisement), an Availability Flag, a Promotion Code, a Bundle Details record (indicating products which are part of a bundle), a Condition Type record (indicating new, used, poor, good, and similar), a Social Rank record (indicating a rank of “likes” and similar of the price), a Votes/Likes record (indicating a number of “likes” and similar which a Price or Product has received), a Price Rank record, a Visibility Indicator record (indicating whether the price is visible to the public, whether it is only visible to a Merchant, or the like), a Supply Chain Reference record (indicating whether the price was obtained from a retailer, a wholesaler, or another party in a supply chain), a Sale Location (indicating a geographic location where the product is available at the price), a Manufactured Location record (indicating where the product was produced or manufactured), a Launch Date record (indicating how long the product has been on the market), and an Age of Product record (indicating how long the product was used by the user). When capitalized herein, the foregoing terms are meant to refer to values in a Price Attribute 340 record.

[Para 34] A Product Attribute 345 record may comprise, for example, values encoding features of or describing a Product. The entire Product Attribute 345 schema may comprise thousands of columns, though only tens or hundreds of the columns may be applicable to any given Product. Product Attributes 345 are described herein and in co-pending application number __, titled “Data Refining Engine for High Performance Analysis System and Method,” and filed contemporaneously herewith. An example set of values in a Product Attribute 345 record for a ring is as follows: Title, “Sterling Silver Diamond & Blue Topaz Ring;” Brand, “Blue Nile;” Category (such as, for example, a Category 335 in a category schema), “rings;” Metal Name, “silver;” Stone Shape, “cushion;” Stone Name, “topaz;” Width, “3 mm;” Stone Color, “blue;” Product Type, “rings,” Birthstone, “September;” and Setting Type, “prong.” An example set of Product Attributes 345 for a shoe is as follows: Brand, “Asics;” Category (such as, for example, a Category 335 in a category schema), “Men’s Sneakers & Athletic;” Shoe Size, “8;” Product Type, “wrestling shoes,” Color, “black;” Shoe Style, “sneakers;” Sports, “athletic;” Upper Material, “mesh.” When capitalized herein, the foregoing terms are meant to refer to values in a Product Attribute 345 record.

[Para 35] As used herein, “Content” comprises text, graphics, images (including still and video images), audio, graphical arrangement, and instructions for graphical arrangement, including HTML and CSS instructions which may, for example, be interpreted by browser applications.

[Para 36] As used herein, a “Listing Page” is a webpage which contains information associated with multiple iPIDs.

[Para 37] As used herein, a “Product Page” is a webpage which contains information associated with a single IPID.

[Para 38] As used herein, “Event” is information generally in news or current events. Events may be found in Content. Listing Pages, Product Pages, and Event Pages are all examples of Webpage Types 350.

[Para 39] Generally, a Crawl Agent 400 obtains Content, such as URI-Content Instance 310, from a webpage served by Web Server 115. The Crawl Agent 400 accesses the webpage via URI 305, which URI is obtained by the Crawl Agent 400 from a URI Queue 355. The URI Queue 355 is maintained by the Indix Server 200 through execution of the Parse Routine 700, the Seeder Routine 800, and the URI Queue Manager Routine 900. The Parse Routine 700 parses URI-Content Instance 310 for price and product information, such as Price Attibutes 340 and Product Attributes 345 (referred to herein together as “Attributes 340/345”), according to a Parse Map 315, and stores the Parse Result 325. The Seeder Routine 800 identifies URIs 305 which contain Attributes 340/345 and adds URIs to the URI Queue 355. The URI Queue Manager Routine 900 adjusts the Time to Next Check 360 of each URI 305 in the URL Queue 355 based on factors such as, for example, analysis of the Parse Result 325 and observed changes in Attributes 340/345 associated with iPIDs 330, whether users searching the Parse Results 325 express interest in an iPID 330, an MPID 332, or a Category 335 associated with the Parse Result 325, the Web Server’s 115 functional or declared availability, changes in a Attributes 340/345 for a Category 335 associated with the iPID 330 (such as a “Price” Attribute), whether the Parse Result 325 is consistent with a Listing Page or a Product Page or another Webpage Type 350, when the URI was last crawled, and according to other criteria.

[Para 40] The Parse Map 315 applied to the URI-Content Instance 310 to convert the URI-Content Instance 310 into Parse Result 325 is selected based on whether there is an existing Parse Map 315 associated with the URI 305, with an iPID 330 associated with the URI 305 (or an Equivalent iPID 334), or associated with a website, domain name, ecommerce platform (as may be provided by Ecommerce Platform 160), or other Parse Map Determiner 320 associated with the URI or URI-Content Instance 310. If there is no Parse Map 315 associated with the URI 305, then one or more general purpose Parse Maps 315 may be selected and used to convert the URI-Content Instance 310 into Parse Result 325. Multiple Parse Maps 315 may be selected, the results thereof screened for data type mis-matches, and the results thereof validated and set as the revised Parse Result 325.

[Para 41] The Parse Map 315 and Parse Result 325 may be validated by human- and machine-based systems. A Parse Map Validation Routine 1000 assists with human-based validation by graphically labeling Parse Results 325 in a webpage or other graphical communication media for human confirmation or correction. The Parse Map Validation Routine 1000 is discussed in relation to Figure 10, while an example of the output of an embodiment of the Parse Map Validation Routine 1000 is illustrated and discussed in relation to Figure 11.

[Para 42] Figure 1 is a network and device diagram illustrating exemplary computing devices configured according to embodiments disclosed in this paper. Illustrated in Figure 1 are an Indix Server 200 and an Indix Database 300. The Indix Server 300 may execute a Cluster Manager Routine 260 to manage clusters of Crawl Agents 400 and clusters of instances of the Indix Server 200. The Indix Server 200 may also execute a Parser Routine 700 to parse a URI-Content Instance 310 into Parse Result 325, discussed further in relation to Figure 7. The Indix Server 200 may also execute a Seeder Routine 800 to add new URIs to the URI Queue 355, discussed further in relation to Figure 8, a URI Queue Manager Routine 900 to manage the URI Queue 355, discussed further in relation to Figure 9, and a Parse Map Validation Routine 1000 to validate Parse Result 325, discussed further in relation to Figures 10 and 11.

[Para 43] The Indix Database 300 is illustrated as comprising an HDFS Database 130, which may store the URI-Content Instance 310 in, for example, a Hadoop distributed file system, and an HBASE Database 135, which may store the Parse Result 325 in, for example, an HBase non-relational distributed database. The Indix Database 300 is discussed further in relation to Figure 3.

[Para 44] Also illustrated in Figure 1 is a Crawl Agent 400, representing Crawl Agents 1 to N, and a Crawl Agent Database 500. The Crawl Agent 400 (including Agents 1 to N) may execute the URI Check Routine 600. The Crawl Agent 400 is discussed further in relation to Figure 4.

[Para 45] Also illustrated in Figure 1 is a Client Device 105, such as a mobile or non-mobile computer device. The Client Device 105 is an example of computing devices such as, for example, a mobile phone, a tablet, laptop, personal computer, gaming computer, or media playback computer. The Client Device 105 represents any computing device capable of rendering Content in a browser or an equivalent user-interface. Client Devices are used by "users." The Client Device 105 may be utilized to search the Parse Results 325 and to validate and improve the Parse Map 315 associated with a URI 305.

[Para 46] Also illustrated in Figure 1 is a Web Server 115, which may serve Content in the form of webpages or equivalent output in response to URIs, such as URI 305.

[Para 47] Also illustrated in Figure 1 is an Ecommerce Platform 160, which may provide ecommerce services, such as website and/or webpage hosting via webpage templates comprising HTML and CSS elements. Customers of Ecommerce Platform 160 may complete the webpage templates with Content and serve the webpages and websites from, for example, Web Server 115.

[Para 48] Interaction among devices illustrated in Figure 1 may be accomplished, for example, through the use of credentials to authenticate and authorize a machine or user with respect to other machines.

[Para 49] In Figure 1, the computing machines may be physically separate computing devices or logically separate processes executed by a common computing device. Certain components are illustrated in Figure 1 as connecting directly to one another (such as, for example, the Indix Database 300 to the Indix Server 200), though the connections may be through the Network 150. If these components are embodied in separate computers, then additional steps may be added to the disclosed invention to recite communicating between the components.

[Para 50] The Network 150 comprises computers, network connections among the computers, and software routines to enable communication between the computers over the network connections. Examples of the Network 150 comprise an Ethernet network, the Internet, and/or a wireless network, such as a GSM, TDMA, CDMA, EDGE, HSPA, LTE or other network provided by a wireless service provider, or a television broadcast facility. Connection to the Network 150 may be via a Wi-Fi connection. More than one network may be involved in a communication session between the illustrated devices. Connection to the Network 150 may require that the computers execute software routines which enable, for example, the seven layers of the OSI model of computer networking or equivalent in a wireless phone network.

[Para 51] This paper may discuss a first computer as connecting to a second computer (such as a Crawl Agent 400 connecting to the Indix Server 200) or to a corresponding datastore (such as to Indix Database 300); it should be understood that such connections may be to, through, or via the other of the two components (for example, a statement that a computing device connects with or sends data to the Indix Server 200 should be understood as saying that the computing device may connect with or send data to the Indix Database 300). References herein to "database" should be understood as equivalent to "datastore." Although illustrated as components integrated in one physical unit, the computers and databases may be provided by common (or separate) physical hardware and common (or separate) logic processors and memory components. Though discussed as occurring within one computing device, the software routines and data groups used by the software routines may be stored and/or executed remotely relative to any of the computers through, for example, application virtualization.

[Para 52] Figure 2 is a functional block diagram of an exemplary Indix Server 200 computing device and some data structures and/or components thereof. The Indix Server 200 in Figure 2 comprises at least one Processing Unit 210, Indix Server Memory 250, a Display 240 and Input 245, all interconnected along with the Network Interface 230 via a Bus 220. The Processing Unit 210 may comprise one or more general-purpose Central Processing Units ("CPU") 212 as well as one or more special-purpose Graphics Processing Units ("GPU") 214. The components of the Processing Unit 210 may be utilized by the Operating System 255 for different functions required by the routines executed by the Indix Server 200. The Network Interface 230 may be utilized to form connections with the Network 150 or to form device-to-device connections with other computers. The Indix Server Memory 250 generally comprises a random access memory ("RAM"), a read only memory ("ROM"), and a permanent mass storage device, such as a disk drive or SDRAM (synchronous dynamic random-access memory).

[Para 53] The Indix Server Memory 250 stores program code for software routines, such as, for example, Cluster Manager Routine 260, Parser Routine 700, Seeder Routine 800, URI Queue Manager Routine 900, and Parse Map Validation Routine 1000, as well as, for example, browser, email client and server routines, client applications, and database applications (discussed further below). Additional data groups for routines, such as for a webserver and web browser, may also be present on and executed by the Indix Server 200 and the other computers illustrated in Figure 1. Webserver and browser routines may provide an interface for interaction among the computing devices, for example, through webserver and web browser routines which may serve and respond to data and information in the form of webpages and html documents or files. The browsers and webservers are meant to illustrate machine- and user-interface and user-interface enabling routines generally, and may be replaced by equivalent routines for serving and rendering information to and in interfaces in a computing device (whether in a web browser or in, for example, a mobile device application).

[Para 54] In addition, the Indix Server Memory 250 also stores an Operating System 255. These software components may be loaded from a non-transient Computer Readable Storage Medium 295 into Indix Server Memory 250 of the computing device using a drive mechanism (not shown) associated with a non-transient Computer Readable Storage Medium 295, such as a floppy disc, tape, DVD/CD-ROM drive, memory card, or other like storage medium. In some embodiments, software components may also or instead be loaded via a mechanism other than a drive mechanism and Computer Readable Storage Medium 295 (e.g., via Network Interface 230).

[Para 55] The computing device 200 may also comprise hardware supporting input modalities, Input 245, such as, for example, a touchscreen, a camera, a keyboard, a mouse, a trackball, a stylus, motion detectors, and a microphone. The Input 245 may also serve as a Display 240, as in the case of a touchscreen display which also serves as Input 245, and which may respond to input in the form of contact by a finger or stylus with the surface of the Input 245.

[Para 56] The computing device 200 may also comprise or communicate via Bus 220 with Indix Datastore 300, illustrated further in Figure 3. In various embodiments, Bus 220 may comprise a storage area network ("SAN"), a high speed serial bus, and/or via other suitable communication technology. In some embodiments, the Indix Server 200 may communicate with the Indix Datastore 300 via Network Interface 230. The Indix Server 200 may, in some embodiments, include many more components than those shown in this Figure. However, it is not necessary that all of these generally conventional components be shown in order to disclose an illustrative embodiment.

[Para 57] Figure 3 is a functional block diagram of the Indix Datastore 300 illustrated in the computing device of Figure 2. The components of the Indix Datastore 300 are data groups used by routines and are discussed further herein in the discussion of other of the Figures. The data groups used by routines illustrated in Figure 3 may be represented by a cell in a column or a value separated from other values in a defined structure in a digital document or file. Though referred to herein as individual records or entries, the records may comprise more than one database entry. The database entries may be, represent, or encode numbers, numerical operators, binary values, logical values, text, string operators, joins, conditional logic, tests, and similar.

[Para 58] Figure 4 is a functional block diagram of an exemplary Crawl Agent 400 computing device and some data structures and/or components thereof. The Crawl Agent 400 in Figure 4 comprises at least one Processing Unit 410, Crawl Agent Memory 450, a Display 440 and Input 445, all interconnected along with the Network Interface 430 via a Bus 420. The Processing Unit 410 may comprise one or more general-purpose Central Processing Units ("CPU") 412 as well as one or more special-purpose Graphics Processing Units ("GPU") 414. The components of the Processing Unit 410 may be utilized by the Operating System 455 for different functions required by the routines executed by the Crawl Agent 400. The Network Interface 430 may be utilized to form connections with the Network 150 or to form device-to-device connections with other computers. The Crawl Agent Memory 450 generally comprises a random access memory ("RAM"), a read only memory ("ROM"), and a permanent mass storage device, such as a disk drive or SDRAM (synchronous dynamic random-access memory).

[Para 59] The Crawl Agent Memory 450 stores program code for software routines, such as, for example, the URI Check Routine 600, as well as, for example, browser, email client and server routines, client applications, and database applications (discussed further below). Additional data groups for routines, such as for a webserver and web browser, may also be present on and executed by the Crawl Agent 400 and the other computers illustrated in Figure 1. Webserver and browser routines may provide an interface for interaction among the computing devices, for example, through webserver and web browser routines which may serve and respond to data and information in the form of webpages and html documents or files. The browsers and webservers are meant to illustrate machine- and user-interface and user-interface enabling routines generally, and may be replaced by equivalent routines for serving and rendering information to and in interfaces in a computing device (whether in a web browser or in, for example, a mobile device application).

[Para 60] In addition, the Crawl Agent Memory 450 also stores an Operating System 455. These software components may be loaded from a non-transient Computer Readable Storage Medium 495 into Crawl Agent Memory 450 of the computing device using a drive mechanism (not shown) associated with a non-transient Computer Readable Storage Medium 495, such as a floppy disc, tape, DVD/CD-ROM drive, memory card, or other like storage medium. In some embodiments, software components may also or instead be loaded via a mechanism other than a drive mechanism and Computer Readable Storage Medium 495 (e.g., via Network Interface 430).

[Para 61] The computing device 400 may also comprise hardware supporting input modalities, Input 445, such as, for example, a touchscreen, a camera, a keyboard, a mouse, a trackball, a stylus, motion detectors, and a microphone. The Input 445 may also serve as a Display 440, as in the case of a touchscreen display which also serves as Input 445, and which may respond to input in the form of contact by a finger or stylus with the surface of the Input 445.

[Para 62] The computing device 400 may also comprise or communicate via Bus 420 with Crawl Agent Datastore 500, illustrated further in Figure 5. In various embodiments, Bus 420 may comprise a storage area network ("SAN"), a high speed serial bus, and/or via other suitable communication technology. In some embodiments, the Crawl Agent 400 may communicate with the Crawl Agent Datastore 500 via Network Interface 430. The Crawl Agent 400 may, in some embodiments, include many more components than those shown in this Figure. However, it is not necessary that all of these generally conventional components be shown in order to disclose an illustrative embodiment.

[Para 63] Figure 5 is a functional block diagram of the Crawl Agent Datastore 500 illustrated in the computing device of Figure 4. The components of the Crawl Agent Datastore 500 are data groups used by routines and are discussed further herein in the discussion of other of the Figures. The data groups used by routines illustrated in Figure 5 may be represented by a cell in a column or a value separated from other values in a defined structure in a digital document or file. Though referred to herein as individual records or entries, the records may comprise more than one database entry. The database entries may be, represent, or encode numbers, numerical operators, binary values, logical values, text, string operators, joins, conditional logic, tests, and similar.

[Para 64] Figure 6 is a flowchart illustrating an embodiment of a URI Check Routine 600 in which the Crawl Agent 400 obtains a URI 305 from a URI Queue 355 and obtains a URI-Content Instance 310. At block 605, the URI Check Routine 600 obtains a URI 305 from, for example, the Indix Server 200 and, for example, the URI Queue Manager Routine 900 and the URI Queue 355. The URIs 305 may be obtained in units comprising more than one URI 305. At block 610, the URI Check Routine 600 utilizes the URI 305 to contact, for example, the Web Server 115 and obtain Content, which Content is referred to herein as a URI-Content Instance 310. At block 615, the URI Check Routine 600 and/or the Web Server 115 transmits the URI-

Content Instance 310 to the Indix Server 300 where it may be stored, for example, in the HDFS Database 130. At block 699, the URI Check Routine 600 may conclude and/or may continue to loop over URIs 305 in the URI Queue 355.

[Para 65] Figure 7 is a flowchart illustrating an embodiment of a Parser Routine 700 for parsing a URI-Content Instance 310 and saving a Parse Result 325. The Parser Routine 700 may be executed by, for example, the Indix Server 200. Blocks 705 to 770 may iterate over each URI-Content Instance 310 in the Indix Datastore 300. At block 710 a decision may be made regarding whether there is a Parse Map 315 associated with the URI 305 associated with the URI-Content Instance 310; the association may be via another record, such as via an iPID 330 (or MPID 332 or other Parse Map Determinor 320) associated with the URI 305, which iPID 330 may be associated with the Parse Map 315. If there is, then at block 715, the associated Parse Map 315 may be obtained. If there is not, then at block 720 a determination may be made regarding whether the URI 305 or the URI-Content Instance 310 is associated with a Parse Map Determiner 320. As above, this association may be via another record, such as via an iPID 330 (or MPID 332) associated with the URI 305, which iPID 330 (or MPID 332) may be associated with the Parse Map Determiner 320.

[Para 66] Parse Map Determiners 320 may be, for example, a Category 335 associated with a URI 305, and/or may be characteristic webpage and/or website stuctures or templates wherein Price and Product Attributes occur in association with specific HTML and CSS elements in the templates. The Parse Map Determiner 320 may comprise, for example, a webpage ecommerce platform (as may be provided by, for example, Ecommerce Platform 160), a store, a merchant, a domain name, and OpenGraph elements in HTML. The association of the URI 305 or the URI-Content Instance 310 with the Parse Map Determiner 320 may be according to a record in the Indix Datastore 300 and/or may be identifiable in the URI-Content Instance 310. If the URI 305 or the URI-Content Instance 310 is associated with a Parse Map Determiner 320, then at block 725 the Parse Map 315 associated therewith may be obtained. If the URI 305 or the URI-Content Instance 310 is not associated with a Parse Map Determiner 320, then at block 730 a generic Parse Map 315 may be obtained. The generic Parse Map 315 may be a Parse Map 315 not associated with a specific URI 305. At block 735, the URI-Content Instance 310 may be parsed according to the Parse Map 315 to create a Parse Result 325. At block 740, the Parse Result 325 may be stored, for example, in the HBase Database 135.

[Para 67] Blocks 750 to 760 may iterate for each of or sets of Attributes 340/345 in the Parse Result 325. At block 750, a determination may be made by the Parser Routine 700 regarding whether there are one or more data type mismatches between expected data types for Attributes 340/345 and the data type of the data stored or attempted to be stored in the Attribute 340/345 in the Parse Result 325. For example, a Price Attribute 340 may be a "Sale Price" which is expected to be a currency amount while the content of the record (or attempted to have been stored in the record) may a date, an image, text, or a value for another non-currency data type. The non-matching data may or may not be stored in the Attribute 340/345, though the attempt to store the non-matching data (and a resulting error message) may be noted and stored in an equivalent record. In relation to a URI-Content Instance 310 the number and type of data type mismatches may be recorded and scored with different weights for different data type mismatches.

[Para 68] If a data type mismatch was determined (or a data type mismatch score above a threshold was calculated) at block 750, then at block 755 the URI 305 associated with the URI-Content Instance 310 may be removed from the URI Queue 355 and, optionally at block 1100, the Parse Map Validation Routine 1000 may be performed relative to the Parse Map 315 utilized to parse the URI-Content Instance 310. If a data type mismatch was not determined (or a data type mismatch score bellow a threshold was calculated) at block 750, then at block 760 the Parser Routine 700 may iterate over the next Attribute 340/345 or set thereof.

[Para 69] At block 765, the memory corresponding to the URI-Content Instance 310 may optionally be labeled to be deleted, recycled, reused, or otherwise discarded. At block 770 the Parser Routine 700 may iterate over the next URI-Content Instance 310. At block 799, the Parser Routine 700 may conclude.

[Para 70] Figure 8 is a flowchart illustrating an embodiment of a Seeder Routine 800 for identifying URIs 305 which contain Price or Product Attributes and adding the URIs to the URI Queue 355. The Seeder Routine 800 may be executed by, for example, the Indix Server 200.

[Para 71] The Seeder Routine 800 may be executed regularly on base URIs already in the Indix system to ensure that all URIs at a particular domain name are being explored.

[Para 72] At block 805 a base URI may be obtained, for example, from the URI 305 records or from an external source of URIs. For each base URI a determination may be made at block 810 whether the URI 305 is associated with a "sitemap" or a "wild" crawl, such as by checking, if available, the Webpage Type 350 record, which record may have been developed previously. For "wild" crawl URIs, at block 815 the Wild Re-Seed Interval may be set while, at block 820, the Sitemap Re-Seed Interval may be set. The Wild Re-Seed Interval may be set to be shorter than the Sitemap Re-Seed Interval. At block 825, the Seeder Routine 800 may return to iterate over the next base URI in a block of base URIs or, for example, the Seeder Routine 800 may proceed to block 830.

[Para 73] At block 830, a base URI either a first time or at its Re-Seed Interval may be obtained. At block 835, the Seeder Routine 800 receives input information such as, for example, a base URI, sample non-Product Pages, sample Product Pages, sample Listing Pages, and sample category home pages (pages listing products according to categories).

[Para 74] At block 840, the Seeder Routine 800 verifies that a Crawl Agent 400 will be allowed to crawl the website by, for example, requesting the Content at, for example, the base URI. At block 845, the Seeder Routine 800 identifies in the Content the site name, site URI, the crawl delay, the contents of a "robots.txt" file, the URI structures for Listing Pages, Product Pages, and non-Product Pages, the depth of Listing and Product Pages from the base URI, deduplication rules (such as invariant query or path parameters), and whether the crawling strategy for the website will be according to a sitemap or whether the crawling strategy will be a "wild crawl," wherein all or substantially all URIs on all pages are identified and followed.

[Para 75] Blocks 850 through 860 may iterate over all URIs 305 identified at block 845. At block 855, the Time of Next Check 360 for the URI 305 may be set in the URI Queue 355. At block 860*, the Seeder Routine 800 may iterate over the remaining URIs 305 identified at block 845. At block 865 the Seeder Routine 800 may conclude iterating over the then-current base URI. At block 899, the Seeder Routine 800 may terminate and/or return to block 830.

[Para 76] Figures 9A and 9B are flowcharts illustrating an embodiment of a URI Queue Manager Routine 900. Blocks 905 through 997 may iterate over each Parse Result 325 stored in the Indix Datastore 300. At block 910 a determination may be made regarding whether the Parse Result 325 comprises an Attribute 340/345. If not, then at block 925, the URI 305 associated with the Parse Result 325 may be removed from the URI Queue 355 and, optionally, the process may then proceed to block 1000 and execution of the Parse Map Validation Routine 1000.

[Para 77] If the determination at block 910 was that the Parse Result 325 comprises an Attribute 340/345, which determination may further comprise determining that the URI 305 is associated with an iPID 330, then Blocks 915 to 930 may iterate for each or sets of Attributes 340/345 in the Parse Result 325. At block 920, a determination may be made by the URI Queue Manager Routine 900 regarding whether there are one or more data type mismatches between expected data types for Attributes 340/345 and the data type of the data stored or attempted to be stored in the Attribute 340/345 in the Parse Result 325.

[Para 78] If a data type mismatch was determined (or a data type mismatch score above a threshold was calculated) at block 920, then at block 925 the URI 305 associated with the URI-Content Instance 310 may be removed from the URI Queue 355 and, optionally at block 1000, the Parse Map Validation Routine 1000 may be performed relative to the Parse Map 315 utilized to parse the URI-Content Instance 310. If a data type mismatch was not determined (or a data type mismatch score bellow a threshold was calculated) at block 920, then at block 930 the URI Queue Manager Routine 900 may iterate over the next Attribute 340/345 or set thereof and/or may proceed to block 935.

[Para 79] At block 935 a determination may be made regarding whether the Parse Result 325 is the same as a Parse Result 325 for a different URI 305 (or a different iPID 330), but the same store. Because of the operation of the Parse Map 315 and the Parser Routine 700, the webpages underlying the two identical Parse Results 325 may have different Content, though the same Attributes 340/345, so they result in the same Parse Results 325. If the same Parse Results 325 are determined at block 935, then at block 940 the URI 305 may be removed from the URI Queue 355 and/or may be labeled as a duplicative Parse Result 325, which effectively means that the two iPIDs 330 are treated as being equivalent; the iPIDs 330 for the URIs 305 may also be labeled as equivalent in, for example, the Equivalent iPID 334 record.

[Para 80] At block 945, which may follow block 935 if no same Parse Result 325 was determined, a determination may be made regarding whether the Parse Result 325 is for a Listing Page or a Product Page or other Webpage Type 350. This determination may be made based on whether more than one product can be identified in the Parse Result 325, in which case it may be classified as a Listing Page. Because of the greater efficiency in processing Listing Pages, Listing Pages may be crawled more frequently. If at block 945 the determination was that the Parse Result 325 was for a Product Page, then at block 955 the Webpage Type 350 of the URI 305 associated with the Parse Result 325 may be stored and the Time to Next Check 360 of the URI 305 may be increased. If at block 945 the determination was that the Parse Result 325 was for a Listing Page, then at block 950 the Webpage Type 350 of the URI 305 associated with the Parse Result 325 may be stored and the Time to Next Check 360 of the URI 305 may be decreased.

[Para 81] At block 960 a determination may be made regarding whether there are HTTP errors in the communication session relating to the Parse Result 325 or whether an Attribute 340/345 in the Parse Result 325 (such as an "Availability" Attribute) includes the text or otherwise indicates "discontinued." If so, then at block 965 the Time to Next Check 360 may be increased.

[Para 82] If not at block 960, then at block 970 a determination may be made regarding whether there is a previous Parse Result 325 for the URI 305. If not, then at block 999 the URI Queue Manager Routine may conclude and/or may return to iterate over the next Parse Result 325.

[Para 83] Turning to Figure 9B, at block 975 a determination may be made regarding whether the Parse Results 325 of block 970 are the same or different, with the result being saved. At block 980 the Change Interval for the Parse Results 325 of 970 may be calculated as the time of the most recent Parse Result 325 minus the time of the earliest Parse Result 325 of block 970, divided by the number of changes between the Parse Results 325. At block 985, the Time to Next Check 360 may be set as the previous Time to Next Check 360 multiplied by one-half of the Change Interval calculated at step 980.

[Para 84] At block 990, a determination may be made regarding whether there has been a change in price in the equivalent Parse Result 325 for products associated with the MPID 332 associated with the URI 305 (if any), which will identify a change in the price for the same product at other stores or merchants. If there has been, then at block 991, the Time to Next Check 360 maybe decreased.

[Para 85] At block 992, a determination may be made regarding whether there has been a change in price in the Parse Results 325 for a Category 335 associated with the iPID 330 or MPID 332. If there has been, then at block 993, the Time to Next Check 360 maybe decreased.

[Para 86] At block 994, a determination may be made regarding whether there has been interest in the Product, iPID 330, MPID 332, or Categories 335 in search queries submitted by Client Devices 105. If there has been, then at block 995, the Time to Next Check 360 maybe decreased.

[Para 87] At block 996 the Time to Next Check 360 may be updated based on factors such as, for example, politeness, such as according to a "robots.txt" or part thereof associated with or found in the Parse Result 325, and/or site traffic for a website associated with the Parse Result 325, which site traffic may be reported by a third party.

[Para 88] At block 997, the URI Queue Manager Routine 900 may iterate over the next Parse Result 325. At block 999, the URI Queue Manager Routine 900 may terminate.

[Para 89] Figure 10 is a flowchart illustrating an embodiment of a Parse Map Validation Routine 1000. Blocks 1005 through 1060 may iterate over URI-Content Instances 310. At block 1010, the Parse Map Validation Routine 1000 may obtain a list of labels for HTML and CSS elements and a list of Attributes 340/345. At block 1015, the Parse Map Validation Routine 1000 may identify in the HTML and CSS elements in the URI-Content Instance 310 the Attributes 340/345 identified by the Parse Map 315 associated with the URI 305 for the URI-Content Instance 310. At block 1020, the Parse Map Validation Routine 1000 may output a graphical representation of the result of block 1015. An example of this is shown in Figure 11.

[Para 90] At block 1025 user feedback may be received from, for example, Client Devices 105. User feedback may be provided by, for example, the user selecting an Attribute 340/345 identified in the graphical output of block 1020, and changing the Attribute 340/345, such as by selecting a different Attribute 340/345 from a drop-down box or the like, which selected Attribute 340/345 may then be associated with the corresponding HTML or CSS element. See, for example, element 1170.

[Para 91] At block 1030 a determination may be made regarding whether the user feedback confirms or changes the Parse Map 315. If the user feedback confirms the Parse Map 315, then at block 1035, the Parse Map 315 associations may be increased via a clustering algorithm. If the user feedback changes the Parse Map 315, then at block 1035, the Parse Map 315 associations may be decreased via a clustering algorithm. The clustering algorithm may group user feedback by HTML and CSS element relative to corresponding Attributes 340/345, with the largest grouping from all the users being assigned to the Parse Map 315. Alternatively, a user or type of user (such as an administrator) may be given more weight than other users, such that such user's association is assigned to the Parse Map 315 regardless of the associations assigned by other users.

[Para 92] At block 1045, the revised Parse Map 315 may be saved and, at block 1050, may be associated with the URI 305. At block 1055 the URI Queue 355 may be updated, for example to add back a URI 305 to the URI Queue 355 which may have been removed, for example, due to a data type mismatch.

[Para 93] Block 1060 indicates a return to iterate over the next URI-Content Instance 310. Block 1099 indicates termination of the Parse Map Validation Routine 1000.

[Para 94] Figure 11 is an illustration of a browser window showing a webpage with HTML and CSS elements corresponding to Attributes being labeled with Attribute names. Element 1100 is a browser window within, for example, a Client Device 105. In this example, the browser window 1100 relates to a single product, namely a bracelet. The browser window comprises a Tab Line 1105, an Address Line 1110, a Top Paragraph 1115, and a Product Box 1175. The Product box 1175 contains information relating to a bracelet.

[Para 95] Within the Product Box 1175, certain components are identified with a solid heavy-line box while other components are identified with a dashed heavy-line box. The heavy-line boxes (dashed or solid) are not present in the native source code for the webpage, but are added by the Parse Map Validation Routine 1000, discussed in Figure 10.

[Para 96] Element 1120 is a heavy-line box drawn around an image frame, the image frame being identified from HTML and/or CSS elements in the source code for the webpage. Within Image Frame 1120 is a Bracelet Image 1125 (in Figure 11, the bracelet is labeled with number 1125, though Bracelet Image 1125 may occupy more of the heavy-line box 1120). Appended to the lower right-hand corner of the Image Frame 1120, in dashed heavy-line box 1150 is text of the Attribute, "Image URL," which indicates that the Parse Map Validation Routine 1000 has identified this portion of the HTML and/or CSS code for the webpage as containing the "image URL" Attribute and has added the text "image URL" to the webpage to identify that this Attribute is associated with this component of the webpage.

[Para 97] Element 1130 is a heavy-line box drawn around a paragraph (or other text container) which is the next paragraph after Image Frame 1120, per the HTML and/or CSS elements in the source code for the webpage. Within Paragraph 1130 is the text "7 ¼" Bracelet in silver," which text is from the source code for the webpage. The HTML source code for the webpage may recite, for example, "<title>7 ¼" Bracelet in silver| Blue Nile</title>." Appended to the Paragaph 1130, in dashed heavy-line box 1135, is text of the Attribute, "Title," which indicates that the Parse Map Validation Routine 1000 has identified this portion of the HTML and/or CSS code for the webpage as containing the “title” Attribute and has added the text “Title” to the webpage to identify that this Attribute is associated with this component of the webpage.

[Para 98] Element 1140 is a heavy-line box drawn around a paragraph (or other text container) which contains the words “In stock,” which text is from the source code for the webpage. Appended to the Paragraph 1140, in dashed heavy-line box 1145, is text of the Attribute, “Availability Text,” which indicates that the Parse Map Validation Routine 1000 has identified this portion of the HTML and/or CSS code for the webpage as containing the “Availability Text” Attribute and has added the text “Availability Text” to the webpage to identify that this Attribute is associated with this component of the webpage. This text may be identified as this Attribute because the words, “In stock” may be associated with this Attribute in the Parse Map Validation Routine 1000.

[Para 99] Element 1155 is a heavy-line box drawn around a paragraph (or other text container) which contains the numbers “1393693,” which number is from the source code for the webpage. Appended to the Paragraph 1155, in dashed heavy-line box 1160, is text of the Attribute, “SKU,” which indicates that the Parse Map Validation Routine 1000 has identified this portion of the HTML and/or CSS code for the webpage as containing the “SKU” Attribute and has added the text “SKU” to the webpage to identify that this Attribute is associated with this component of the webpage. This text may be identified as this Attribute because the words, “Item #” followed by a number may be associated with this Attribute in the Parse Map Validation Routine 1000.

[Para 100] Element 1165 is a heavy-line box drawn around a paragraph (or other text container) which contains the currency value “$89.99,” which currency value is from the source code for the webpage. Appended to the Paragraph 1165 is dashed heavy-line box 1170 containing text of several Attributes, in this example, “Price,” “Sale Price,” and “Rebate.” In this embodiment, which is provided as an example, this indicates that the Parse Map Validation Routine 1000 has identified this portion of the HTML and/or CSS code for the webpage as containing the “Price” Attribute. The HTML may recite, for example, “<div class="strong">Price:</div><div class="value"> $89.99 </div>.” In this embodiment, a user has selected box 1170 and activated a list of alternative Attributes (which may be selected by the Parse Map Validation Routine 1000 because they match the data type of the data in box 1165 and sent to the Client Device). In this embodiment, the user may select one of the alternative Attributes from this list, which selection may be transmitted back to, for example, the Parse Map Validation Routine 1000, as discussed above in relation to Figure 10.

[Para 101] Figure 12 is a flowchart illustrating an embodiment of an MPID Assigner Routine 1200. Blocks 1205 to 1260 iterate for successive Parse Results 325. At block 1210, for a first pass of the MPID Assigner Routine 1200, a set of machine-learning algorithms 1-N are executed relative to the Parse Result 325 to classify the Parse Result 325 within the Category 335 taxonomy based on, for example, the URI 305. The different algorithms have different criteria and may or may not produce the same result. For example, one algorithm may place the Parse Result 325 in a Category 335 such as "Tools & Hardware > Tools > Hand Tools > Pliers" while another algorithm may place the Parse Result 325 in a Category 335 such as "Tools & Hardware > Tools > Hand Tools > Screw Drivers." For this first pass, the results of the classification algorithms 1-N may be sent to a testing and validation process, which may include screening for data-type mismatches and/or human review of the results. The testing and validation process may return a ranking of the results by each algorithm. At box 1215, in a subsequent (not first) pass the classification algorithm selected at step 1210 may be executed. If the selected algorithm fails, then the next classification algorithm by rank may be selected and performed relative to the Parse Result 325. The output is a Category 335 assigned to the Parse Result 325.

[Para 102] At box 1220, for a first pass of the MPID Assigner Routine 1200, the MPID Assigner Routine 1200 may execute Attribute 340/345 extraction algorithms 1-M on the Parse Result 325. The extraction algorithms 1-M may be selected based on the Category 335 assigned at step 1215. Similar to above, the results of the attribute extraction algorithms 1-M may be sent to a testing and validation process, which may include screening for data-type mismatches and/or human review of the results. The testing and validation process may return a ranking of the results by each algorithm. At box 1225, in a subsequent (not first) pass the attribute extraction algorithm selected at step 1220 may be executed. If the selected algorithm fails, then the next classification algorithm by rank may be selected and performed relative to the Parse Result 325.

[Para 103] At box 1230, the extracted Attributes 340/345 may be weighted, for example, to weigh certain Attributes 340/345, such as product codes or webpage titles, more heavily than other Attributes 340/345. There may be, for example, four weight factors assigned to each of the Attributes 340/345.

[Para 104] At box 1235, the MPID Assigner Routine 1200 may get the Attributes 340/345 of all iPIDs 330 in the Category 335 assigned at box 1215. If not already weighted (as in step 1230), the MPID Assigner Routine 1200 may weight the Attributes 340/345 according to, for example, the four weight factors.

[Para 105] At box 1240, the MPID Assigner Routine 1200 may cluster the IPIDs 330 in the Category 335 based on the weighted Attributes 340/345 of boxes 1230 and 1235. Because certain Attributes 340/345 are weighted more heavily than others, a match between product codes (such as a UPC number) for two different iPIDs 330 will likely result in the Attribute 340/345 cluster locating the two different iPIDS 330 proximate to one another. Clustering may proceed through progressively smaller clusters until the cluster size grows too small, for example, if each cluster contains just two or one iPIDs 330, at which point the MPID Assigner Routine 1200 may stop clustering and may "back off" one or two cluster steps, until the cluster size is no longer too small.

[Para 106] At box 1245, the MPID Assigner Routine 1200 may identify the iPID 330 in the Category 335 in each cluster which has the maximum number of Attributes 340/345 in common with other iPIDs 330 in that particular level of the Category 335 taxonomy and in that cluster. At box 1250 the MPID Assigner Routine 1200 may assign the iPID 330 identified at box 1245 as the MPID 332 for all the iPIDs 330 in that particular level of the Category 335 taxonomy and in that cluster. At box 1255, the extracted attributes (not weighted) may be saved as Price Attributes 340 and Product Attributes 345 with the MPID 332 assigned at box 1250 and with the Category 335 assigned at step 1215. At box 1260 the MPID Assigner Routine 1200 may return to iterate over the next Parse Result 325 and, at box 1299, the MPID Assigner Routine 1200 may end.

[Para 107] The above Detailed Description of embodiments is not intended to be exhaustive or to limit the disclosure to the precise form disclosed above. While specific embodiments of, and examples are described above for illustrative purposes, various equivalent modifications are possible within the scope of the system, as those skilled in the art will recognize. For example, while processes or blocks are presented in a given order, alternative embodiments may perform routines having operations, or employ systems having blocks, in a different order, and some processes or blocks may be deleted, moved, added, subdivided, combined, and/or modified. While processes or blocks are at times shown as being performed in series, these processes or blocks may instead be performed in parallel, or may be performed at different times. Further, any specific numbers noted herein are only examples; alternative implementations may employ differing values or ranges.

CLAIMS

Claim 1. A computer implement method of obtaining information from a webserver, the method comprising:

obtaining a first URI from a prioritized URI queue;

utilizing the first URI at a first URI access time to request first content from the webserver;

parsing the first content a first time for first price and product information and saving the result as a first parse result;

utilizing the first URI at a second URI access time to request second content from the webserver;

parsing the second content for second price and product information, and saving the result as a second parse result; and

determining that the first parse result is different than the second parse result and setting a time for accessing the first URI in the prioritized URI queue based on the difference.

Claim 2. The method according to Claim 1, wherein determining that the first parse result is different than the second parse result further comprises determining a change interval.

Claim 3. The method according to Claim 2, wherein the change interval is calculated by subtracting the time-date of the later parse result from the time-date of the earlier parse result and dividing this by the number of changes.

Claim 4. The method according to Claim 3, wherein setting the time for accessing the first URI in the prioritized URI queue based on the difference comprises adding one-half the product of the change interval to the time of the most recent URI access time and setting this result as the time for accessing the first URI.

Claim 5. The method according to Claim 1, wherein parsing the first content a first time for first price and product information further comprises selecting a first parse map, which first parse map maps the first content to the first price and product information.

Claim 6. The method according to Claim 5, wherein selecting the first parse map comprises determining that the first URI is associated with a parse map and selecting the associated parse map as the first parse map.

Claim 7. The method according to Claim 5, wherein selecting the first parse map comprises determining that the first URI is not associated with a parse map, determining that the first URI is associated with a parse map determiner and selecting as the first parse map a parse map associated with the parse map determiner, wherein the parse map determiner is one of a group comprising a category associated with the URI, a webpage ecommerce platform, a store, a domain name, and OpenGraph elements in HTML.

Claim 8. The method according to Claim 5, wherein selecting the first parse map comprises determining that the first URI is not associated with a parse map, determining that the first URI is not associated with a parse map determiner, and selecting as the first parse map a generic parse map.

Claim 9. The method according to Claim 5, further comprising a first list comprising HTML and CSS elements, a second list comprising price and product attributes and wherein the parse map comprises an association between at least a first element of the first list and at least a first attribute of the second list.

Claim 10. The method according to Claim 1, wherein:

a first computer obtains the first URI from the prioritized URI queue, utilizes the first URI at the first URI access time to request the first content from the webserver, utilizes the first URI at the second URI access time to request the second content from the webserver;

a second computer parses the first content the first time for first price and product information and saves the result as the first parse result, parses the second content for second price and product information, and saves the result as the second parse result; and

a third computer determines that the first parse result is different than the second parse result and sets the time for accessing the first URI in the prioritized URI queue based on the difference.

Claim 11. A computer implemented method of classifying a first webpage containing information regarding a product and grouping the first webpage with prior webpages containing information regarding the product, the method comprising:

obtaining a first parse result comprising a first set of price and product information parsed from the first webpage and a first identifier associated with the first webpage;

utilizing at least a first algorithm to determine a category for the first webpage from a category taxonomy;

utilizing at least a second algorithm to extract a first set of price and product attributes from the first parse result;

obtaining prior sets of price and product attributes and prior identifiers for other webpages associated with the determined category;

weighing at least one of the product attributes in the first set of price and product attributes and in the prior sets of price and product attributes heavier than other of the attributes;

clustering the weighted price and product attributes to identify webpages with similar price and product attributes;

identifying within each cluster a set of price and product attributes which shares the maximum number of weighted price and product attributes with the other sets of price and product attributes; and

assigning the identifier associated with the set of price and product attributes which shares the maximum number of weighted price and product attributes with the other sets of price and product attributes as a common identifier for all of the products in the cluster.

Claim 12. A computer implemented method of obtaining information from a webserver, the method comprising:

obtaining a first URI from a prioritized URI queue;

utilizing the first URI at a first URI access time to request first content from the webserver;

parsing the first content a first time for first price and product information and saving the result as a first parse result; and

determining that the first parse result does not contain price and product information and removing the first URI from the prioritized URI queue.

Claim 13. A computer implemented method of obtaining information from a webserver, the method comprising:

obtaining a first URI from a prioritized URI queue;

utilizing the first URI at a first URI access time to request first content from the webserver;

parsing the first content a first time for first price and product information and saving the result as a first parse result; and

determining whether the first parse result contains a listing webpage or a product webpage; and

if the first parse result contains a listing webpage, reducing the time to the next URI check of the first URI in the prioritized URI queue; else

increasing the time to the next URI check of the first URI in the prioritized URI queue.

Claim 14. A computer implemented method of obtaining information from a webserver, the method comprising:

obtaining a first URI from a prioritized URI queue;

utilizing the first URI at a first URI access time to request first content from the webserver;

parsing the first content a first time for first price and product information according to a first parse map and saving the result as a first parse result;

determining that a data type of a price or product attribute in the parse result does not match an allowed data type; and

validating the parse map.

Claim 15. A computer implemented method of determining a parse map for parsing price and product information from first content obtained via a first URI, the method comprising:

obtaining a first list comprising HTML and CSS elements;

obtaining a second list comprising price and product attributes, which attributes are each associated with a label;

associating at least a first element of the first list with at least a first attribute of the second list, which association is a first parse map;

obtaining first content via a first URI, which first content comprises HTML and CSS elements;

modifying the first content to graphically identify the portion of the first content encompassed by the first element with the label associated with the first attribute; and

transmitting to a second computer the modified first content.

Claim 16. The method according to Claim 15, further comprising receiving at least one instruction to associate the first element with a second attribute of the second list, associating the first element with the second attribute, removing the association between first element and the first attribute, and associating the first element and the second attribute in a second parse map.

Claim 17. The method according to Claim 16, further comprising receiving at least two instructions to associate the first element with a second attribute of the second list prior to associating the first element with the second attribute, removing the association between first element and the first attribute, and associating the first element and the second attribute in a second parse map.

Claim 18. The method according to Claim 16, further comprising receiving multiple instructions to associate the first element with different attributes of the second list, determining via clustering which among the different attributes of the second list is most often instructed to be associated with the first element, and associating the attribute of the second list which is most often instructed to be associated with the first element as the second attribute, removing the association between first element and the first attribute, and associating the first element and the second attribute in a second parse map.

Claim 19. The method according to Claim 15, further comprising:

determining that the first URI is associated with a parse map determiner, which parse map determiner is a group comprising a webpage ecommerce platform, a store, a domain name, and OpenGraph elements in HTML; and wherein

the first parse map is a parse map associated with the parse map determiner.

Claim 20. The method according to Claim 15, wherein modifying the first content to graphically identify the portion of the first content encompassed by the first element with the label associated with the first attribute further comprises:

providing the second computer with a selectable list of attribute labels;

receiving from the second computer a selection of an attribute label from the selectable list; and

associating the first element with the attribute corresponding to the selected attribute label as a second attribute, removing the association between first element and the first attribute, and associating the first element and the second attribute in a second parse map.

Claim 21. A method of adding URIs to a URI queue, practiced by a first computer comprising a memory, the method comprising:

with the first computer, receiving a base URI and sample non-product webpages, sample product pages, sample listing webpages, and sample category webpages associated with the base URI;

with the first computer, verifying that the first computer is allowed to crawl a website accessed via the base URI and downloading content from the website;

with the first computer, identifying in the content at least one of a site name, a crawl delay, URI structures associated with the listing pages, product pages, and non-product pages, URI deduplication rules for the website;

with the first computer, determining a crawling strategy as at least one of a sitemap-based crawling strategy or a wild crawl based crawling strategy; and for each URI identified thereby,

adding the identified URI to a URI queue and setting a time to next check the identified URI.

Claim 22. A computing apparatus for obtaining information from a webserver, the apparatus comprising a processor and a memory storing instructions that, when executed by the processor, configure the apparatus to:

obtain a first URI from a prioritized URI queue;

utilize the first URI at a first URI access time to request first content from the webserver;

parse the first content a first time for first price and product information and save the result as a first parse result;

utilize the first URI at a second URI access time to request second content from the webserver;

parse the second content for second price and product information, and save the result as a second parse result; and

determine that the first parse result is different than the second parse result and set a time for accessing the first URI in the prioritized URI queue based on the difference.

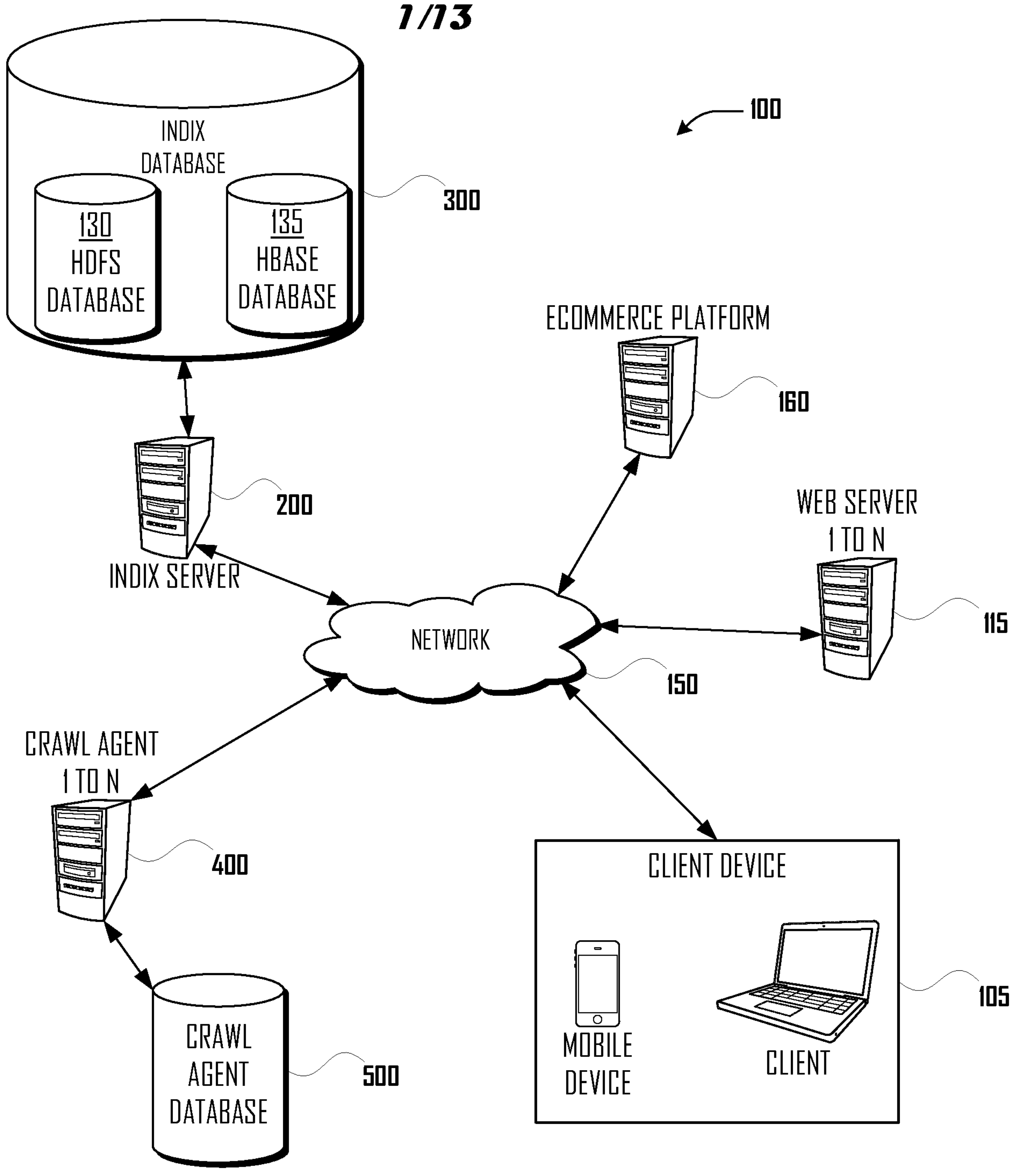
100
INDIX DATABASE
130 HDFS DATABASE
135 HBASE DATABASE
300
ECOMMERCE PLATFORM
160
200
INDIX SERVER
WEB SERVER 1 TO N
115
NETWORK
150
CRAWL AGENT 1 TO N
400
CLIENT DEVICE
MOBILE DEVICE
CLIENT
105
CRAWL AGENT DATABASE
500

Fig.1

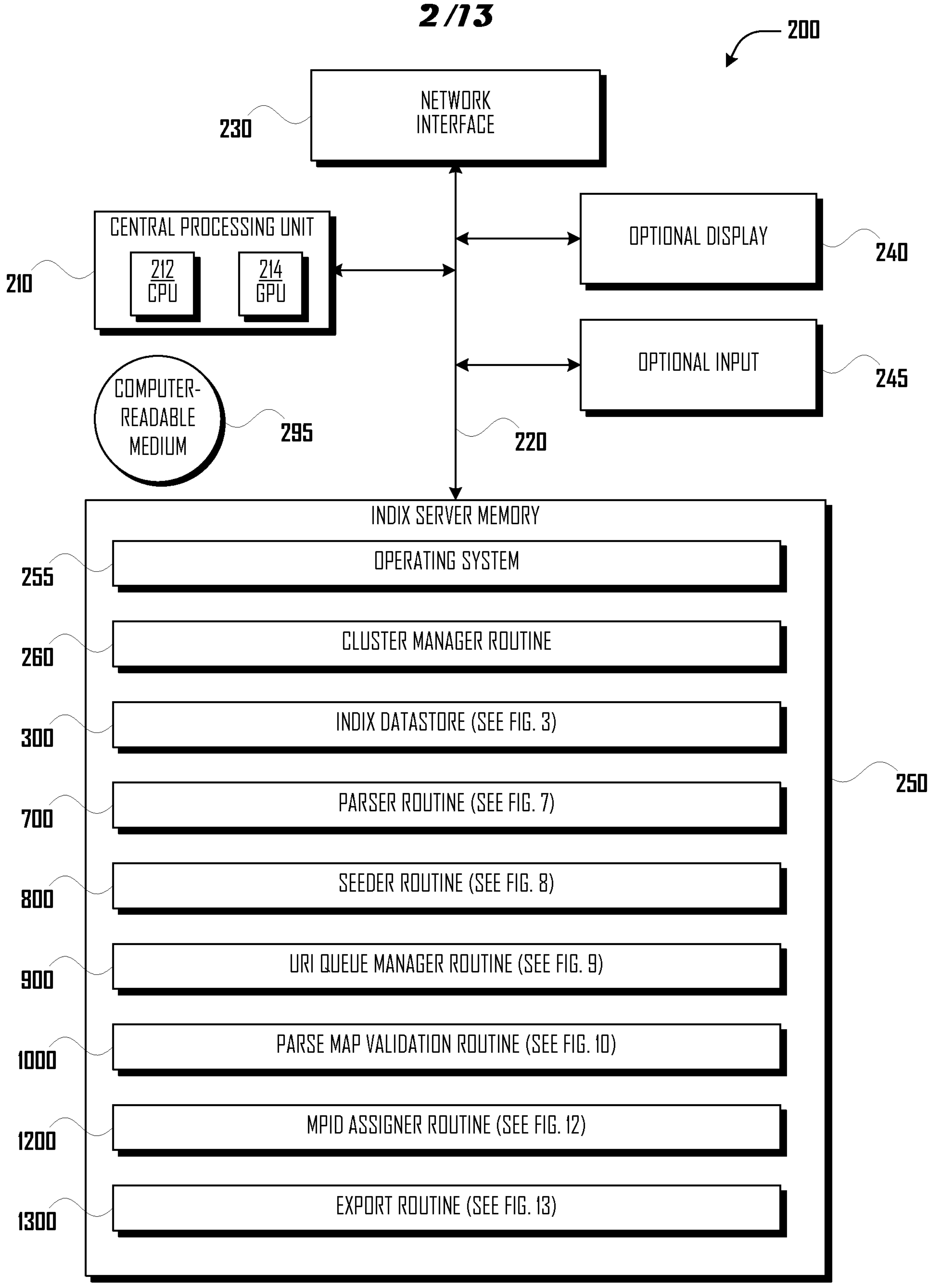
200
NETWORK INTERFACE
230
CENTRAL PROCESSING UNIT
212 CPU
214 GPU
210
OPTIONAL DISPLAY
240
OPTIONAL INPUT
245
COMPUTER-READABLE MEDIUM
295
220
INDIX SERVER MEMORY
255 OPERATING SYSTEM
260 CLUSTER MANAGER ROUTINE
300 INDIX DATASTORE (SEE FIG. 3)
700 PARSER ROUTINE (SEE FIG. 7)
800 SEEDER ROUTINE (SEE FIG. 8)
900 URI QUEUE MANAGER ROUTINE (SEE FIG. 9)
1000 PARSE MAP VALIDATION ROUTINE (SEE FIG. 10)
1200 MPID ASSIGNER ROUTINE (SEE FIG. 12)
1300 EXPORT ROUTINE (SEE FIG. 13)
250

Fig. 2

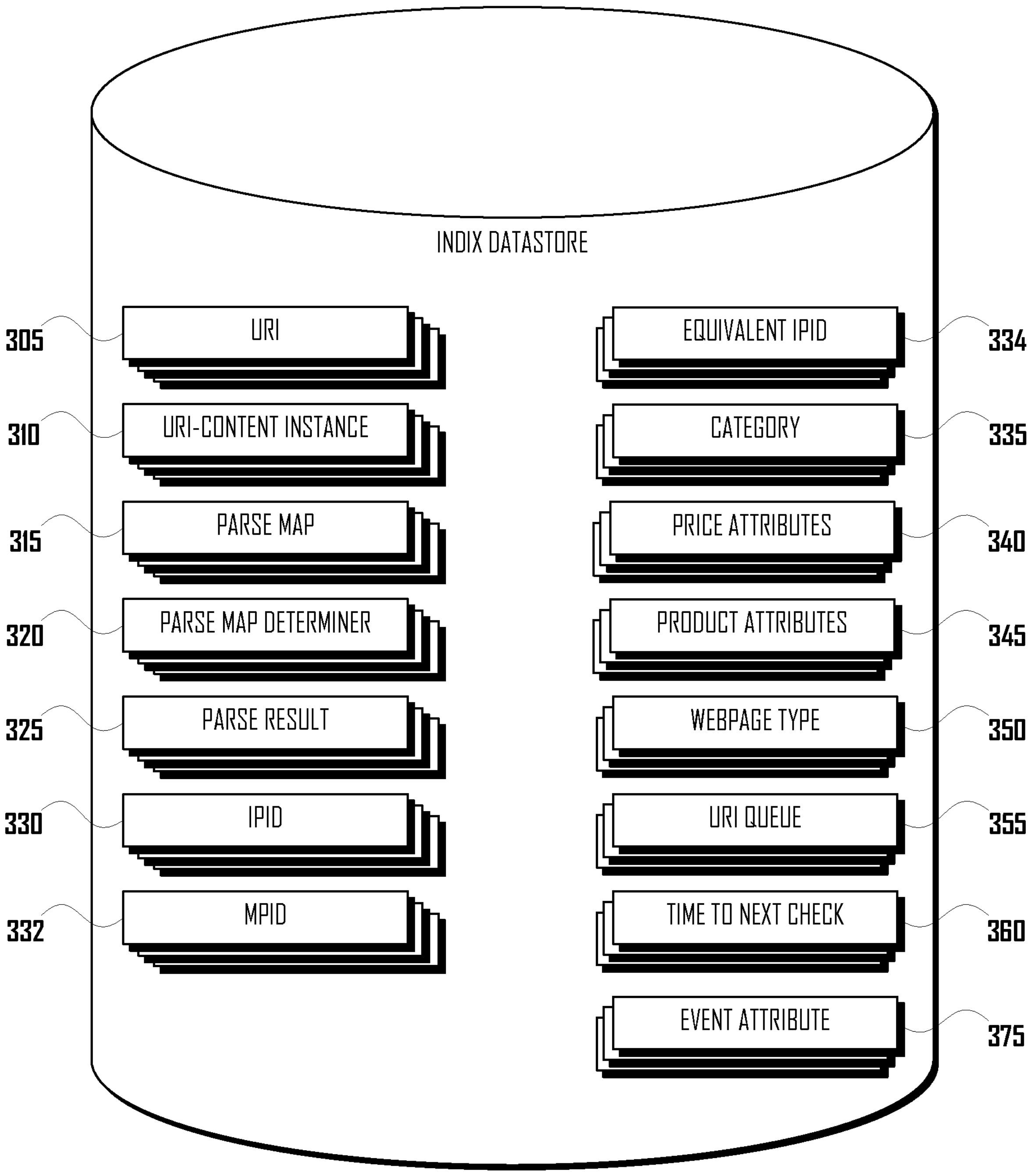
INDIX DATASTORE
305 URI
310 URI-CONTENT INSTANCE
315 PARSE MAP
320 PARSE MAP DETERMINER
325 PARSE RESULT
330 IPID
332 MPID
334 EQUIVALENT IPID
335 CATEGORY
340 PRICE ATTRIBUTES
345 PRODUCT ATTRIBUTES
350 WEBPAGE TYPE
355 URI QUEUE
360 TIME TO NEXT CHECK
375 EVENT ATTRIBUTE

Fig.3

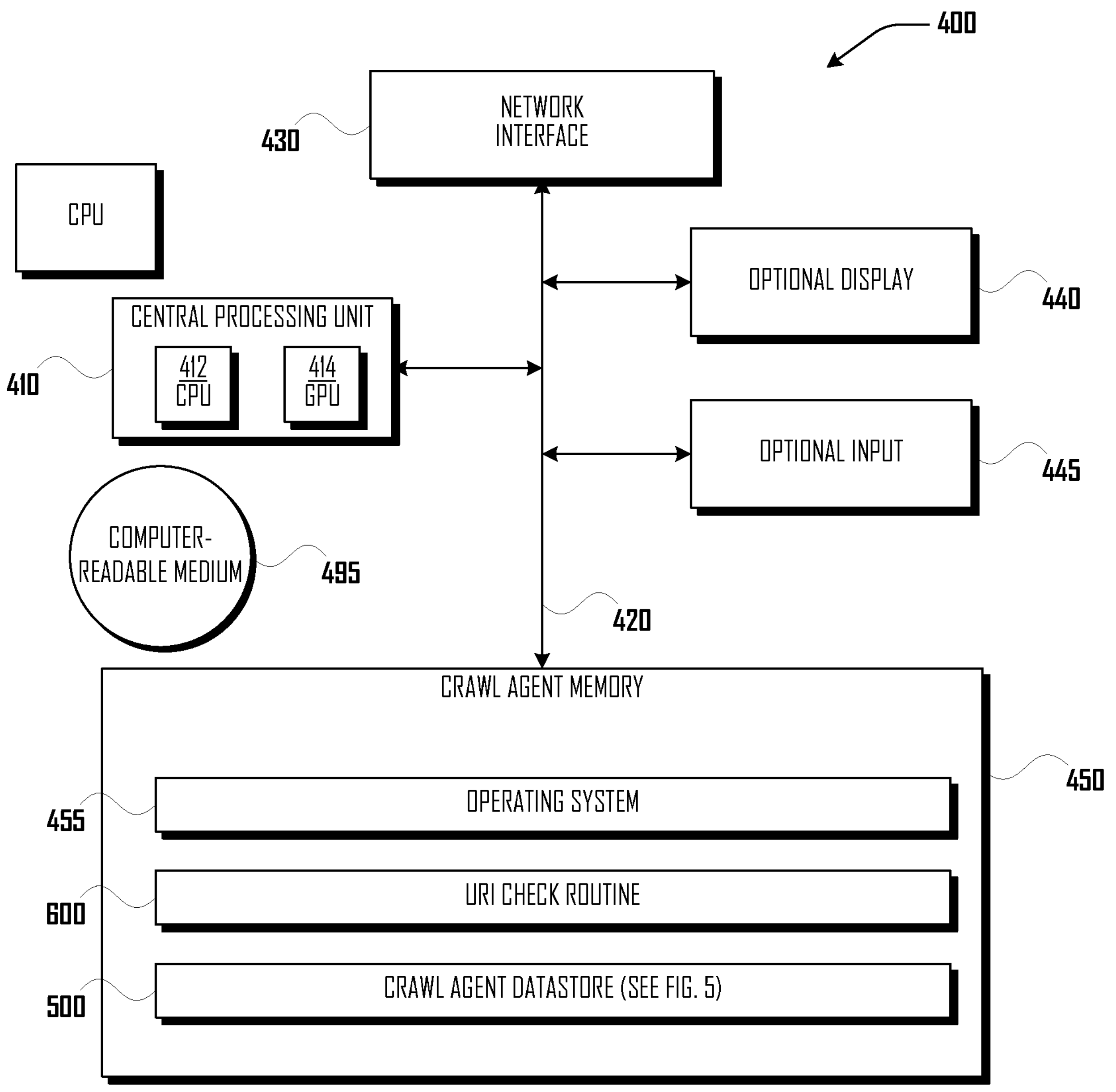
400
NETWORK INTERFACE
430
CPU
CENTRAL PROCESSING UNIT
412 CPU
414 GPU
410
OPTIONAL DISPLAY
440
OPTIONAL INPUT
445
COMPUTER-READABLE MEDIUM
495
420
CRAWL AGENT MEMORY
450
OPERATING SYSTEM
455
URI CHECK ROUTINE
600
CRAWL AGENT DATASTORE (SEE FIG. 5)
500

Fig.4

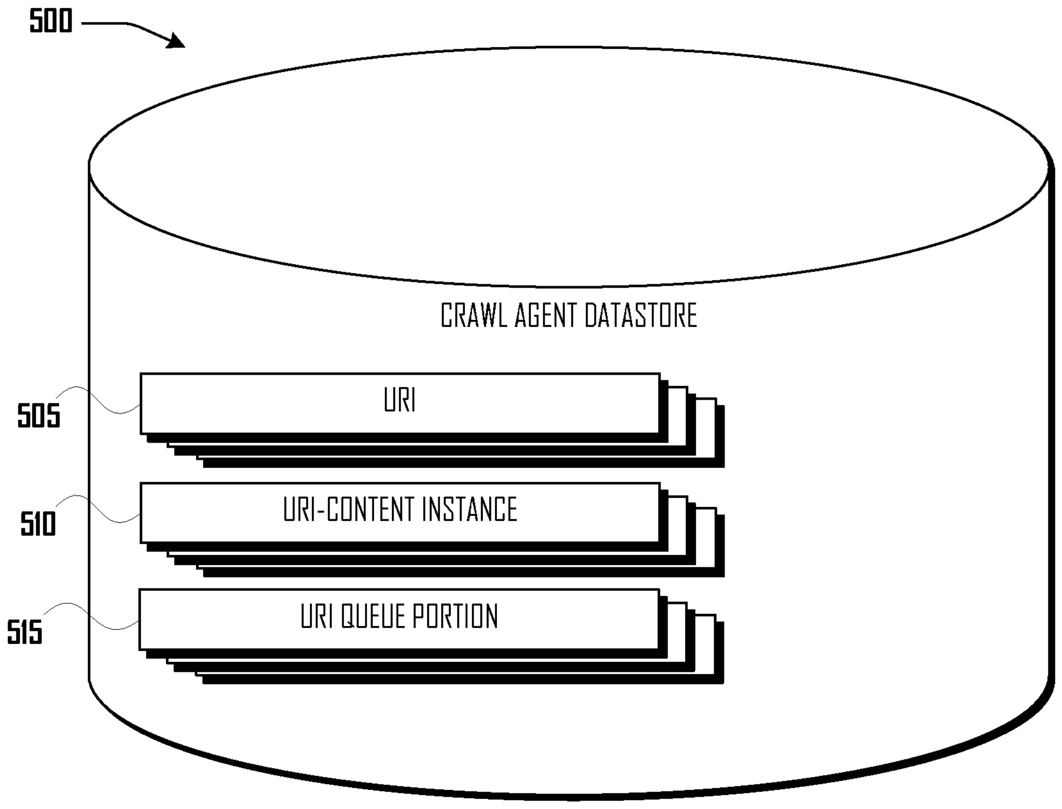
500
CRAWL AGENT DATASTORE
505
URI
510
URI-CONTENT INSTANCE
515
URI QUEUE PORTION

Fig.5

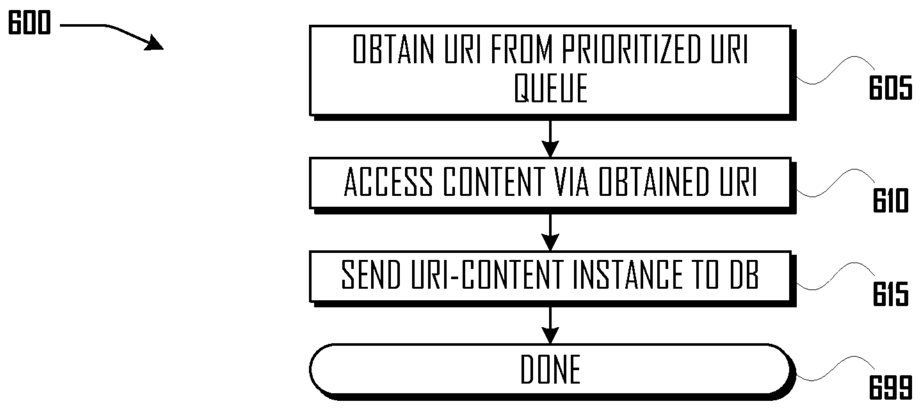
600
OBTAIN URI FROM PRIORITIZED URI QUEUE
605
ACCESS CONTENT VIA OBTAINED URI
610
SEND URI-CONTENT INSTANCE TO DB
615
DONE
699

Fig.6

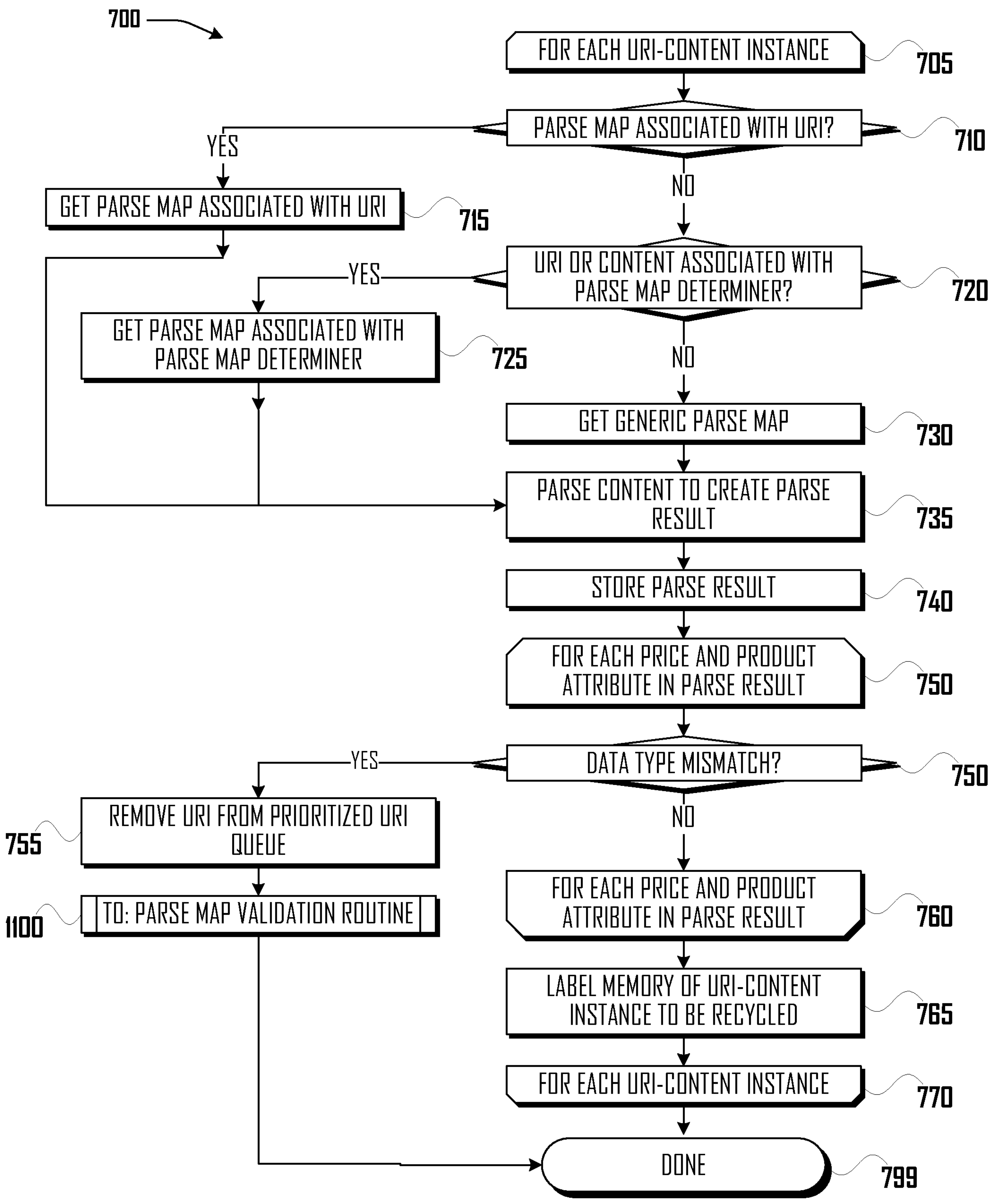
700
FOR EACH URI-CONTENT INSTANCE
705
PARSE MAP ASSOCIATED WITH URI?
710
YES
GET PARSE MAP ASSOCIATED WITH URI
715
NO
URI OR CONTENT ASSOCIATED WITH PARSE MAP DETERMINER?
720
YES
GET PARSE MAP ASSOCIATED WITH PARSE MAP DETERMINER
725
NO
GET GENERIC PARSE MAP
730
PARSE CONTENT TO CREATE PARSE RESULT
735
STORE PARSE RESULT
740
FOR EACH PRICE AND PRODUCT ATTRIBUTE IN PARSE RESULT
750
DATA TYPE MISMATCH?
750
YES
REMOVE URI FROM PRIORITIZED URI QUEUE
755
TO: PARSE MAP VALIDATION ROUTINE
1100
NO
FOR EACH PRICE AND PRODUCT ATTRIBUTE IN PARSE RESULT
760
LABEL MEMORY OF URI-CONTENT INSTANCE TO BE RECYCLED
765
FOR EACH URI-CONTENT INSTANCE
770
DONE
799

Fig.7

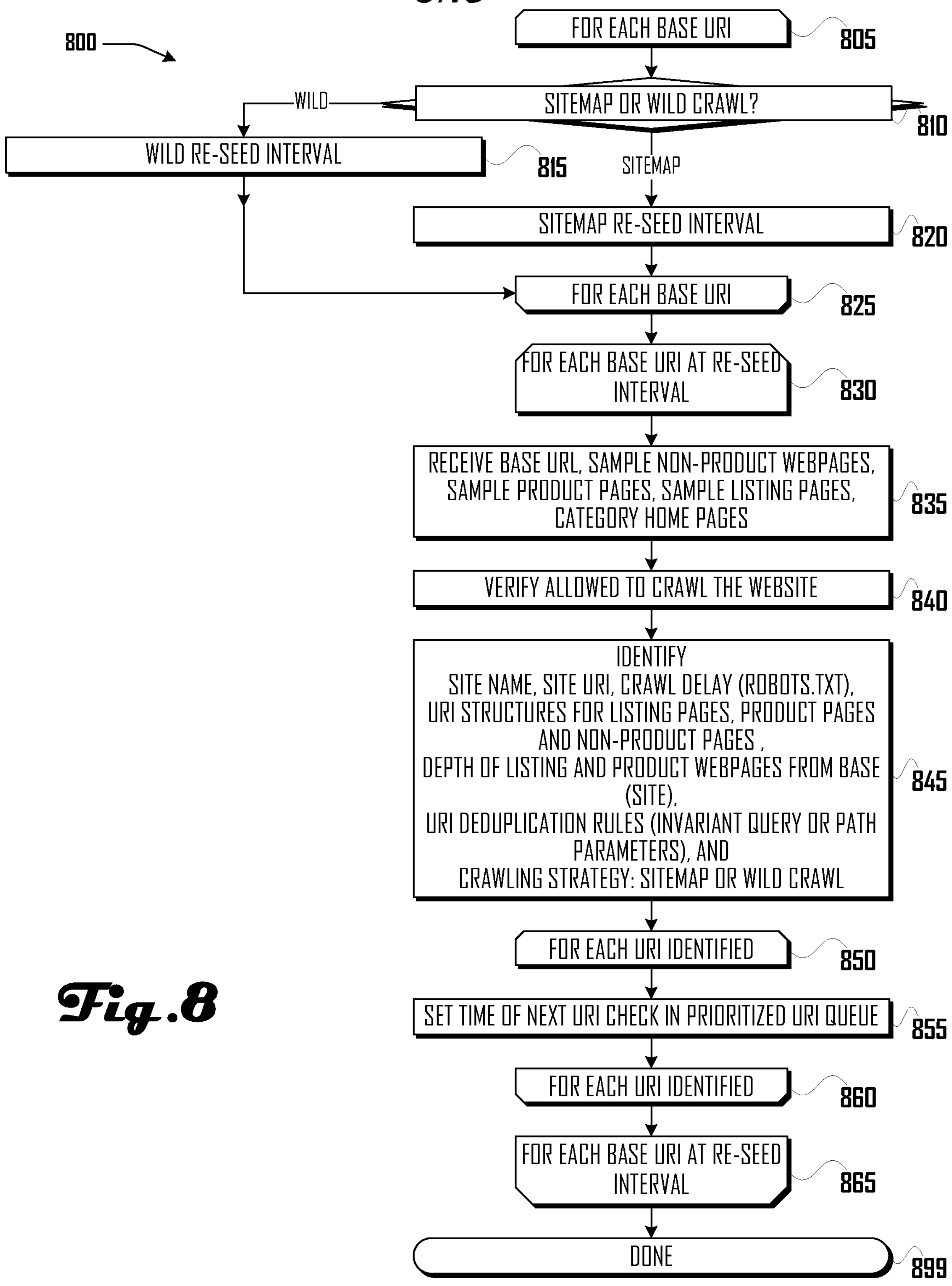
800
FOR EACH BASE URI
805
SITEMAP OR WILD CRAWL?
810
WILD
WILD RE-SEED INTERVAL
815
SITEMAP
SITEMAP RE-SEED INTERVAL
820
FOR EACH BASE URI
825
FOR EACH BASE URI AT RE-SEED INTERVAL
830
RECEIVE BASE URL, SAMPLE NON-PRODUCT WEBPAGES, SAMPLE PRODUCT PAGES, SAMPLE LISTING PAGES, CATEGORY HOME PAGES
835
VERIFY ALLOWED TO CRAWL THE WEBSITE
840
IDENTIFY SITE NAME, SITE URI, CRAWL DELAY (ROBOTS.TXT), URI STRUCTURES FOR LISTING PAGES, PRODUCT PAGES AND NON-PRODUCT PAGES , DEPTH OF LISTING AND PRODUCT WEBPAGES FROM BASE (SITE), URI DEDUPLICATION RULES (INVARIANT QUERY OR PATH PARAMETERS), AND CRAWLING STRATEGY: SITEMAP OR WILD CRAWL
845
FOR EACH URI IDENTIFIED
850
SET TIME OF NEXT URI CHECK IN PRIORITIZED URI QUEUE
855
FOR EACH URI IDENTIFIED
860
FOR EACH BASE URI AT RE-SEED INTERVAL
865
DONE
899

Fig.8

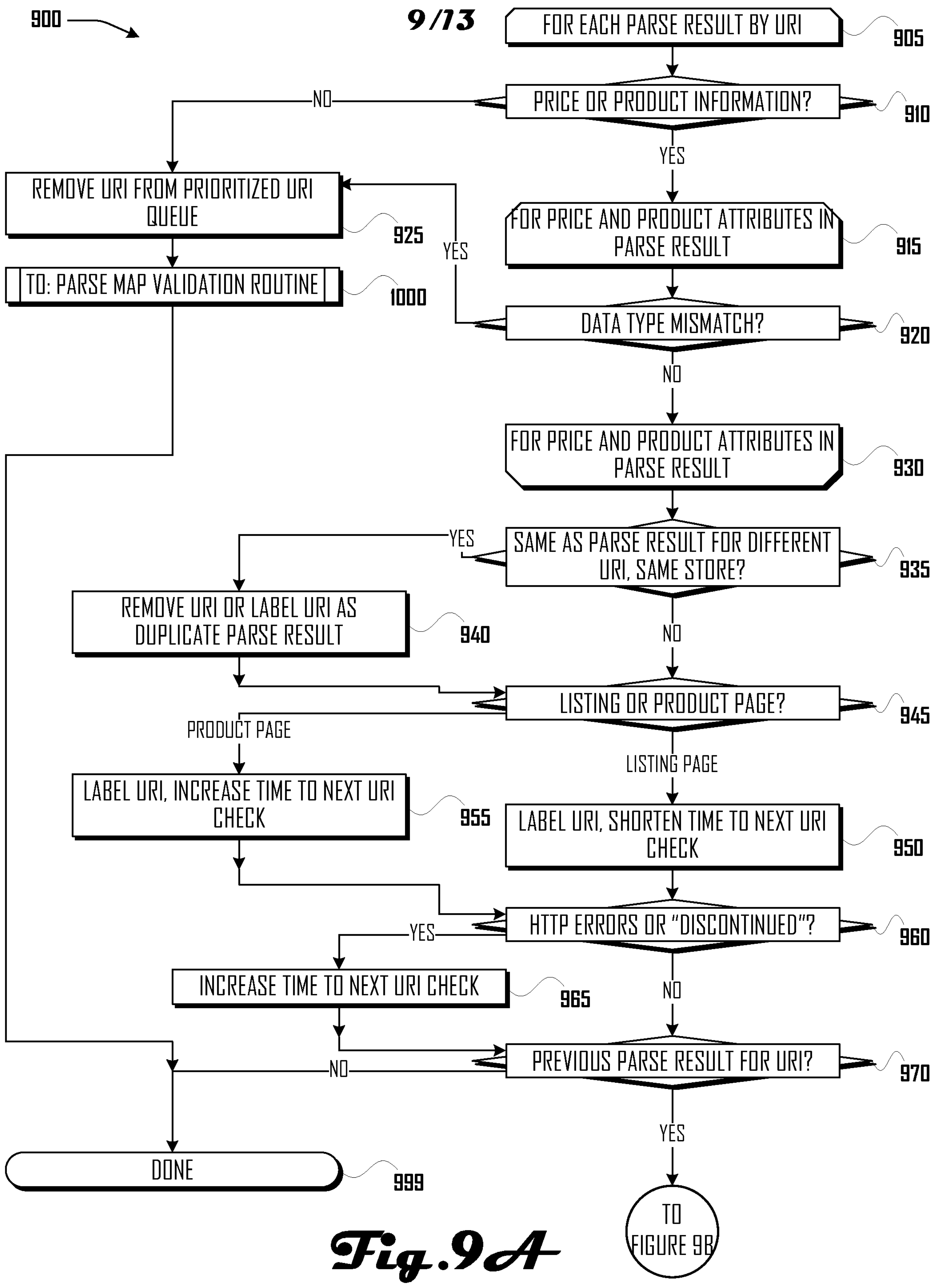
900
FOR EACH PARSE RESULT BY URI
905
PRICE OR PRODUCT INFORMATION?
910
NO
YES
REMOVE URI FROM PRIORITIZED URI QUEUE
925
TO: PARSE MAP VALIDATION ROUTINE
1000
FOR PRICE AND PRODUCT ATTRIBUTES IN PARSE RESULT
915
YES
DATA TYPE MISMATCH?
920
NO
FOR PRICE AND PRODUCT ATTRIBUTES IN PARSE RESULT
930
YES
SAME AS PARSE RESULT FOR DIFFERENT URI, SAME STORE?
935
REMOVE URI OR LABEL URI AS DUPLICATE PARSE RESULT
940
NO
LISTING OR PRODUCT PAGE?
945
PRODUCT PAGE
LISTING PAGE
LABEL URI, INCREASE TIME TO NEXT URI CHECK
955
LABEL URI, SHORTEN TIME TO NEXT URI CHECK
950
HTTP ERRORS OR "DISCONTINUED"?
960
YES
INCREASE TIME TO NEXT URI CHECK
965
NO
PREVIOUS PARSE RESULT FOR URI?
970
NO
YES
DONE
999
TO FIGURE 9B

Fig.9A

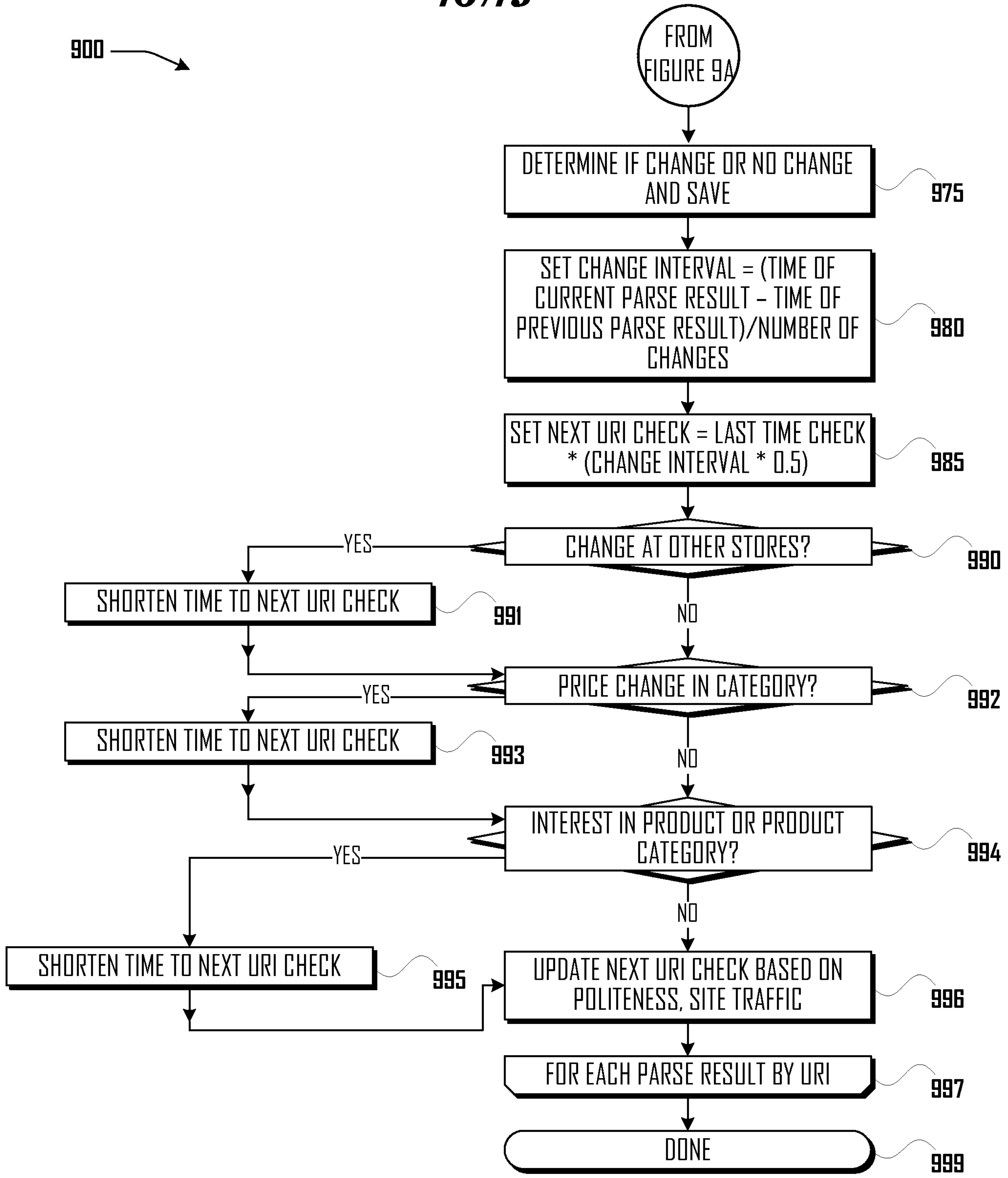
900
FROM FIGURE 9A
DETERMINE IF CHANGE OR NO CHANGE AND SAVE
975
SET CHANGE INTERVAL = (TIME OF CURRENT PARSE RESULT – TIME OF PREVIOUS PARSE RESULT)/NUMBER OF CHANGES
980
SET NEXT URI CHECK = LAST TIME CHECK * (CHANGE INTERVAL * 0.5)
985
CHANGE AT OTHER STORES?
990
YES
SHORTEN TIME TO NEXT URI CHECK
991
NO
PRICE CHANGE IN CATEGORY?
992
YES
SHORTEN TIME TO NEXT URI CHECK
993
NO
INTEREST IN PRODUCT OR PRODUCT CATEGORY?
994
YES
SHORTEN TIME TO NEXT URI CHECK
995
NO
UPDATE NEXT URI CHECK BASED ON POLITENESS, SITE TRAFFIC
996
FOR EACH PARSE RESULT BY URI
997
DONE
999

Fig. 9B

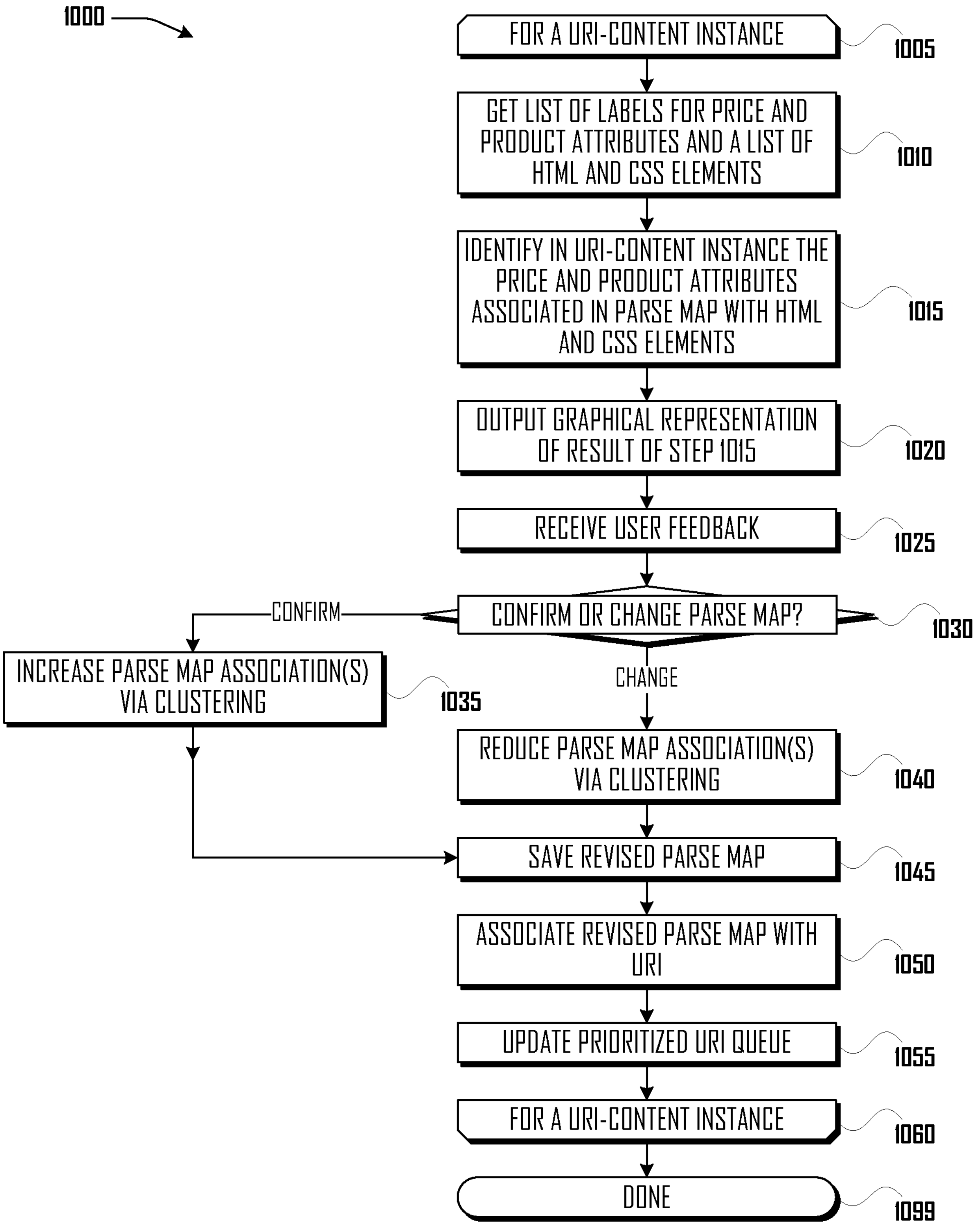
1000
FOR A URI-CONTENT INSTANCE
1005
GET LIST OF LABELS FOR PRICE AND PRODUCT ATTRIBUTES AND A LIST OF HTML AND CSS ELEMENTS
1010
IDENTIFY IN URI-CONTENT INSTANCE THE PRICE AND PRODUCT ATTRIBUTES ASSOCIATED IN PARSE MAP WITH HTML AND CSS ELEMENTS
1015
OUTPUT GRAPHICAL REPRESENTATION OF RESULT OF STEP 1015
1020
RECEIVE USER FEEDBACK
1025
CONFIRM OR CHANGE PARSE MAP?
1030
CONFIRM
INCREASE PARSE MAP ASSOCIATION(S) VIA CLUSTERING
1035
CHANGE
REDUCE PARSE MAP ASSOCIATION(S) VIA CLUSTERING
1040
SAVE REVISED PARSE MAP
1045
ASSOCIATE REVISED PARSE MAP WITH URI
1050
UPDATE PRIORITIZED URI QUEUE
1055
FOR A URI-CONTENT INSTANCE
1060
DONE
1099

Fig.10

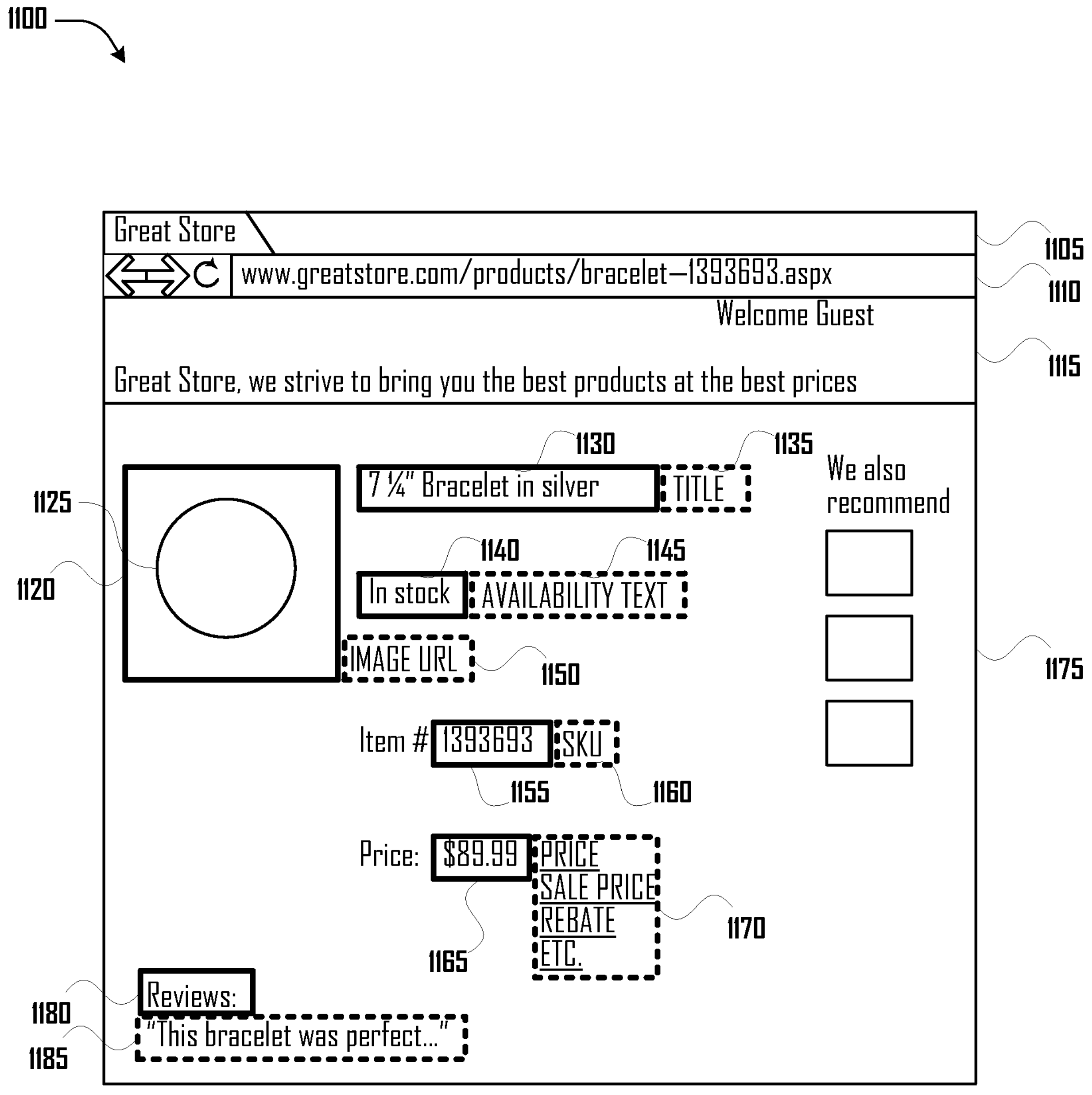
1100
Great Store
www.greatstore.com/products/bracelet–1393693.aspx
Welcome Guest
Great Store, we strive to bring you the best products at the best prices
7 ¼" Bracelet in silver
TITLE
We also recommend
In stock
AVAILABILITY TEXT
IMAGE URL
Item #
1393693
SKU
Price:
$89.99
PRICE
SALE PRICE
REBATE
ETC.
Reviews:
"This bracelet was perfect..."
1105
1110
1115
1120
1125
1130
1135
1140
1145
1150
1155
1160
1165
1170
1175
1180
1185

Fig.11

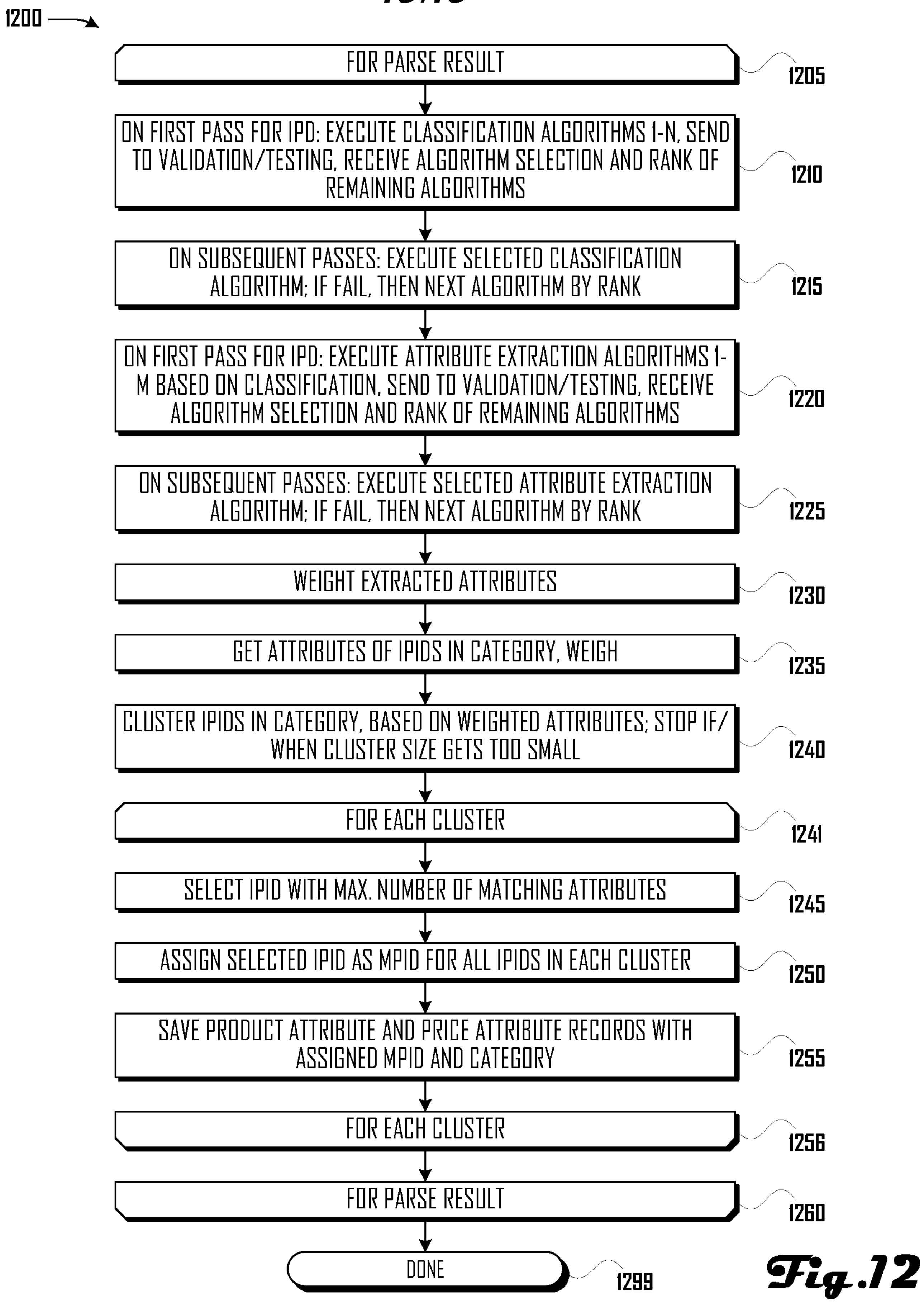
1200
FOR PARSE RESULT
1205
ON FIRST PASS FOR IPD: EXECUTE CLASSIFICATION ALGORITHMS 1-N, SEND TO VALIDATION/TESTING, RECEIVE ALGORITHM SELECTION AND RANK OF REMAINING ALGORITHMS
1210
ON SUBSEQUENT PASSES: EXECUTE SELECTED CLASSIFICATION ALGORITHM; IF FAIL, THEN NEXT ALGORITHM BY RANK
1215
ON FIRST PASS FOR IPD: EXECUTE ATTRIBUTE EXTRACTION ALGORITHMS 1-M BASED ON CLASSIFICATION, SEND TO VALIDATION/TESTING, RECEIVE ALGORITHM SELECTION AND RANK OF REMAINING ALGORITHMS
1220
ON SUBSEQUENT PASSES: EXECUTE SELECTED ATTRIBUTE EXTRACTION ALGORITHM; IF FAIL, THEN NEXT ALGORITHM BY RANK
1225
WEIGHT EXTRACTED ATTRIBUTES
1230
GET ATTRIBUTES OF IPIDS IN CATEGORY, WEIGH
1235
CLUSTER IPIDS IN CATEGORY, BASED ON WEIGHTED ATTRIBUTES; STOP IF/WHEN CLUSTER SIZE GETS TOO SMALL
1240
FOR EACH CLUSTER
1241
SELECT IPID WITH MAX. NUMBER OF MATCHING ATTRIBUTES
1245
ASSIGN SELECTED IPID AS MPID FOR ALL IPIDS IN EACH CLUSTER
1250
SAVE PRODUCT ATTRIBUTE AND PRICE ATTRIBUTE RECORDS WITH ASSIGNED MPID AND CATEGORY
1255
FOR EACH CLUSTER
1256
FOR PARSE RESULT
1260
DONE
1299

Fig.12

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2013/052106

A. CLASSIFICATION OF SUBJECT MATTER

G06F 17/00(2006.01)i

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)
G06F 17/00; G06F 15/16; G06F 17/26; G06F 15/173; G06F 17/30; G06Q 30/00; G06F 17/60; G06F 7/00; H04L 9/32; G06F ; G06F 12/14

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords:price, product parse, URI, URN, URL, content, element, classify, category, attribute, queue, list, and similar terms.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2011-0087647 A1 (ALESSIO SIGNORINI et al.) 14 April 2011 See paragraphs [0008]-[0009], [0021]-[0022], [0026]-[0027]. [0031], [0038], [0046], [0051]-[0052], [0057]-[0058], [0062], [0065], and [0069]-[0072]; figures 1, 2B-3, 5, and 7; and claim 1.	1-10,12-14,22
A	KR 10-2010-0037401 A (NAVER CORPORATION) 09 April 2010 See paragraphs [0007], [0009]-[0017], and [0022]-[0024]; and figure 1.	1-10,12-14,22
A	US 7,711,775 B2 (MICHAEL A. TAVIS et al.) 04 May 2010 See column 2, line 64 - line 3, line 8; and column 9, line 45 - column 10, line 24; and figure 4.	1-10,12-14,22
A	US 2010-0211474 A1 (PAUL C. GRAHAM et al.) 19 August 2010 See paragraphs [0041]-[0046]; and figure 4.	1-10,12-14,22
A	WO 2002-010961 A2 (ZILLIANT, INC.) 07 February 2002 See page 13, lines 13-28; page 20, lines 1-25, and page 23, line 10 - page 24, line 32; and figure 1.	11
A	WO 2002-035421 A1 (NETSCAPE COMMUNICATIONS CORPORATION) 02 May 2002 See page 6, line 8 - page 7, line 2; page 11, lines 29-35; and page 13, lines 24-31; figures 7-8; and claim 1.	11

☒ 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 26 December 2013 (26.12.2013)	Date of mailing of the international search report **26 December 2013 (26.12.2013)**
Name and mailing address of the ISA/KR Korean Intellectual Property Office 189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701, Republic of Korea Facsimile No. +82-42-472-7140	Authorized officer NHO, Ji Myong Telephone No. +82-42-481-8528

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2007-0043723 A1 (ELAN BITAN et al.) 22 February 2007 See paragraphs [0101], [0104], and [0303]-[0307]; and figures 3 and 5.	11
A	WO 2005-036306 A2 (CNET NETWORKS, INC.) 21 April 2005 See paragraphs [0002]-[0003], [0007], [0033], [0039]-[0044], [0064]-[0068], and [0119]; figures 5, 7, and 9; and claim 1.	15-20
A	US 2004-0168124 A1 (MICHAEL BEISIEGEL et al.) 26 August 2004 See paragraphs [0007]-[0016], [0076]-[0081], and [0087]-[0088]; and figures 2-4.	15-20
A	US 6,381,597 B1 (SIMON M. LIN) 30 April 2002 See column 4, line 65 - column 5, line 65; and column 7, lines 8-35; and figures 4 and 7.	15-20
A	US 2001-0047404 A1 (TAKASHI SUDA) 29 November 2001 See paragraphs [0006]-[0018], [0042], [0046]-[0052], and [0075]-[0089]; and figures 2 and 8.	21
A	US 2006-0075500 A1 (JUSTIN R. BERTMAN et al.) 06 April 2006 See paragraphs [0009] and [0026]-[0027]; and figure 2.	21
A	US 2007-0276816 A1 (JOHN T. SAMPLE et al.) 29 November 2007 See paragraphs [0004], [0038]-[0042], and [0049]; and figure 2.	21

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

Ⅰ. Claims 1-10, 12-14, and 22 relate to a computer implemented method and a computing apparatus of obtaining information from a webserver.
Ⅱ. Claim 11 relates to a computer iplemented method of classfying a first webpage containing information regarding a product grouping the first webpage with prior webpages containing information regarding the produt.
Ⅲ. Claims 15-20 relate to a computer implemented method of determining a parse map for parsing price and product information from first content obtained bia a first URI.
Ⅳ. Claim 21 relates to a method of adding URIs to a URI queue.

1. ☐ As all required addtional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. ☒ As all searchable claims could be searched without effort justifying an additional fees, this Authority did not invite payment of any additional fees.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.
☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.
☐ No protest accompanied the payment of additional search fees.

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2011-0087647 A1	14/04/2011	None	
KR 10-2010-0037401 A	09/04/2010	KR 10-1074578 B1	17/10/2011
US 7711775 B2	04/05/2010	US 2003-0084138 A1	01/05/2003
		WO 2003-036474 A1	01/05/2003
US 2010-0211474 A1	19/08/2010	US 8438076 B2	07/05/2013
		WO 2010-093858 A1	19/08/2010
WO 2002-010961 A2	07/02/2002	AU 2001-079010 A1	13/02/2002
WO 2002-035421 A1	02/05/2002	AU 1231501 A	06/05/2002
US 2007-0043723 A1	22/02/2007	US 2006-0218146 A1	28/09/2006
		US 2011-0055185 A1	03/03/2011
		US 7617193 B2	10/11/2009
WO 2005-036306 A2	21/04/2005	AU 2001-059400 A1	20/11/2001
		AU 2003-234693 A1	27/10/2003
		CA 2378175 A1	15/11/2001
		EP 1287419 A2	05/03/2003
		EP 1493107 A1	05/01/2005
		EP 1639505 A2	29/03/2006
		EP 1668538 A2	14/06/2006
		EP 1668538 A4	03/10/2007
		JP 2005-522784 A	28/07/2005
		JP 2006-527886 A	07/12/2006
		US 2003-0065643 A1	03/04/2003
		US 2004-0078284 A1	22/04/2004
		US 2004-0143600 A1	22/07/2004
		US 2006-0242192 A1	26/10/2006
		US 2012-0233170 A1	13/09/2012
		US 6535880 B1	18/03/2003
		US 6714933 B2	30/03/2004
		US 6725222 B1	20/04/2004
		US 7082426 B2	25/07/2006
		WO 2001-086377 A3	28/02/2002
		WO 2003-088086 A1	23/10/2003
		WO 2005-001595 A2	06/01/2005
		WO 2005-001595 A3	09/09/2005
		WO 2005-036306 A3	13/10/2005
US 2004-0168124 A1	26/08/2004	CA 2349905 A1	07/12/2002
		CN 1313953 C	02/05/2007
		CN 1513145 A	14/07/2004
		EP 1399841 A1	24/03/2004
		JP 2004-530225 A	30/09/2004
		KR 10-0583517 B1	24/05/2006

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
		WO 2002-101579 A1	19/12/2002
US 6381597 B1	30/04/2002	AU 2001-014932 A1	10/05/2001
		CN 1408093 A	02/04/2003
		TW 501033 A	01/09/2002
		WO 2001-026018 A2	12/04/2001
		WO 2001-026018 A3	08/08/2002
US 2001-0047404 A1	29/11/2001	EP 1158427 A2	28/11/2001
		EP 1158427 A3	18/02/2004
		JP 2002-049541 A	15/02/2002
US 2006-0075500 A1	06/04/2006	EP 1834243 A2	19/09/2007
		US 2006-0075468 A1	06/04/2006
		US 2006-0075490 A1	06/04/2006
		US 2006-0075494 A1	06/04/2006
		US 7287279 B2	23/10/2007
		WO 2006-039351 A2	13/04/2006
		WO 2006-039351 A3	30/04/2009
		WO 2006-099282 A2	21/09/2006
		WO 2006-099282 A3	10/01/2008
US 2007-0276816 A1	29/11/2007	US 2010-0088308 A1	08/04/2010
		US 7685133 B2	23/03/2010
		US 8024318 B2	20/09/2011