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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 February 2001 (08.02.2001)

PCT

(10) International Publication Number
WO 01/09773 A1

(51) International Patent Classification[7]: G06F 17/30

(21) International Application Number: PCT/US00/07071

(22) International Filing Date: 17 March 2000 (17.03.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/146,081 28 July 1999 (28.07.1999) US
09/505,261 16 February 2000 (16.02.2000) US

(71) Applicant: **EWONDERS.COM** [US/US]; 9 Whippany Road, Whippany, NJ 07981 (US).

(72) Inventors: **JAFREY, Ali**; 98 Joann Ct., Monmouth Junction, NJ 08852 (US). **JETLEY, Manu**; 19 Amelia Street, North Caldwell, NJ 07006 (US).

(74) Agent: **FORTKORT, Michael, P.**; Mayer, Fortkort & Williams, Suite 250, 200 Executive Drive, West Orange, NJ 07052 (US).

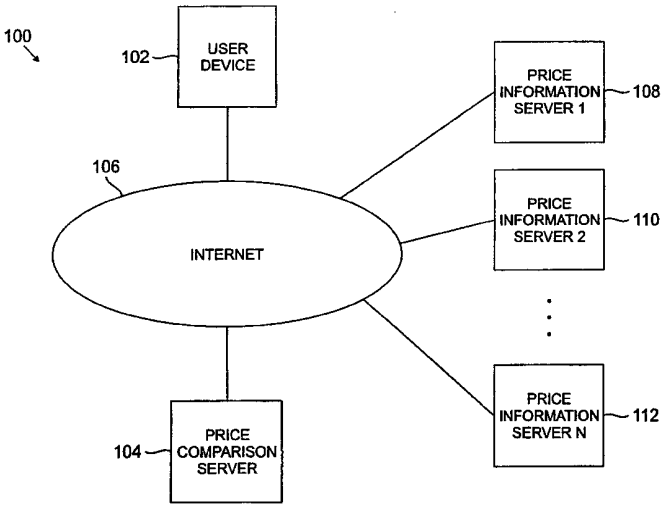
(81) Designated States *(national)*: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *With international search report.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: DYNAMIC DATA GATHERING MARKUP LANGUAGE



(57) Abstract: The embodiments of the present invention solve this problem by providing a programming language for use in specifying a query to a web site (106) and for returning the results of the query in a predetermined manner. One embodiment of the invention comprises a method and apparatus for updating pricing information (108 and 110) over a network (106). A request for pricing information is received at a first network device (108). The request for pricing information is sent ot a second network device (110). A first set of pricing information is received in a dynamic data gathering markup language (DDGML) format.

WO 01/09773 A1

DYNAMIC DATA GATHERING MARKUP LANGUAGE

STATEMENT OF RELATED APPLICATION

This application claims the benefit of prior filed co-pending U.S. Provisional Patent Application Serial No. 60/146081, filed July 28, 1999.

FIELD OF THE INVENTION

The embodiments of the present invention relate generally to programming languages, and more particularly, to a programming language for use in querying a web site on the World Wide Web (WWW) of the Internet and transferring a response to the query to the requesting site in real time.

BACKGROUND OF THE INVENTION

On the WWW portion of the Internet there are a growing number of web sites that present various types of comparisons to the users. One of the most important categories of such web sites are the web sites that provide price comparisons to shoppers. Given the vast amount of information available on the Internet, these price comparison web sites provide extremely useful services. These web sites essentially accept a product and give the user a list of on-line stores that carry this product. along with the current price of the product. These sites basically provide the user with a one-stop research facility where a user can do their price comparison and then go on to buy

the product from whichever store offers the cheapest price.

Up until now most of these web sites have been driven by data feeds provided by the on-line stores. Participating stores provide the comparison web site with a data dump of products for which they wish to provide comparisons along with the updated price per product. Once a user searches for that specific product they are able to see the price of that product being offered by the store. Often, the user can rank the results of his search by price.

One problem that can occur while using this type of an implementation is that the data feeds are static. Once a data feed has been received and has been incorporated into the search engine, the price cannot be updated without updating the data feed. Data feeds are usually updated once a day; thus, for a hot selling product on which price could really be changed more than once a day to keep ahead of the competition, such as computers, the user might not always see the most updated price. For example, being the seller of the absolutely cheapest computer resulting from a comparison on one of these web sites is an enviable position, as the seller in this position will obtain more hits, i.e., Internet shoppers that visit the seller's web site, (and hopefully, as a result, more sales) than those sellers in other positions on the list. A seller may be willing to cut its price, even by a couple of dollars to maintain its position in the list. Yet, the once per day data dumps from web sites to the search engines means that one cannot adjust one's price until the next day, potentially missing many potential hits or sales.

To tackle this problem a new breed of web sites has emerged. These web sites perform the search in real time and provide the user with the most recent prices. These web sites use data parsing techniques to provide the price comparison. Usually a Hypertext Transport Protocol (HTTP) request is made to the target page, the Hypertext Markup Language (HTML) commands contained within the target page is retrieved and then parsed to get to the price for a specific product. This approach requires a very high level of research per site for which the price is retrieved, thereby making such an approach impractical for applications involving many web sites. Moreover, if the layout of the target web site changes, updates need to be made to the parsing engine to retrieve the prices. Apart from constant updates, this approach is very bandwidth intensive. The

complete HTML page or a large portion of it has to be retrieved to extract the price. Some web sites do not even provide the price for the products in clear text; they use American Standard Code for Information Interchange (ASCII) values to display the price so that their web sites are protected from such attempts thereby minimizing network traffic.

The present invention is directed to the problem of developing a method and apparatus for querying of a web site, or multiple web sites, without knowing the format of the web site as a prerequisite and without requiring large bandwidth data transfers, yet also returning the data in real time.

SUMMARY OF THE INVENTION

The embodiments of the present invention solve this problem by providing a programming language for use in specifying a query to a web site and for returning the results of the query in a predetermined manner. One embodiment of the invention comprises a method and apparatus for updating pricing information over a network. A request for pricing information is received at a first network device. The request for pricing information is sent to a second network device. A first set of pricing information is received in a dynamic data gathering markup language (DDGML) format.

With these and other advantages and features of the invention that will become hereinafter apparent, the nature of the invention may be more clearly understood by reference to the following detailed description of the invention, the appended claims and to the several drawings attached herein.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates a system suitable for practicing one embodiment of the invention.

FIG. 2 is a block diagram of a price comparison server in accordance with one embodiment of the invention.

FIG. 3 is a block diagram of a price information server in accordance with one embodiment of the invention.

FIG. 4 is a block flow diagram of the steps performed by a price comparison server in accordance with one embodiment of the invention.

FIG. 5 is a block flow diagram of the steps performed by a price information server in accordance with one embodiment of the invention.

DETAILED DESCRIPTION

The embodiments of the invention comprise a method and apparatus for retrieving pricing information from a web site in real time while minimizing the bandwidth necessary to communicate the pricing information. One embodiment of the invention includes a price comparison server and a price information server. The price comparison server is equipped with a price comparison module and a price update module. The price information server includes a unique and customized smart agent. In operation, the price comparison server receives a user command to gather current pricing information for a product or service. The price comparison server formulates a query for current pricing information and sends the query to one or more price information storage servers. Each price information server receives the query, gathers the current pricing information for the requested product or service, formats the current pricing information in a compact and efficient manner, and sends the current pricing information to the price comparison server. The price comparison server then receives and processes the current pricing information, and then presents the current pricing information to the user in a desired format.

The embodiments of the invention utilize a new programming language referred to as the Dynamic Data Gathering Markup Language (DDGML). DDGML is an application language that is constructed using the Extensible Markup Language (XML). By having both the price comparison server and price information server transferring information using a predetermined format (i.e., DDGML), the price information server can provide near real time pricing information to a user without having the user wait an undue amount of time for such information.

According to the embodiments of the present invention, DDGML collects, filters and presents real-time information without the limitations of web site boundaries.

Moreover, DDGML is adopted to acquire advantage in E-commerce. DDGML is an interactive tool that benefits both the online shopper and the online merchant by furnishing Comparison Interactive (CI) messaging and Search/Discover Discipline (SDD). CI messaging presents real-time pricing information to the online shopper, and a highly robust SDD helps online merchants learn the competition and adjust their business in real-time.

Traditional price comparison web sites are deficient because web site content is populated by way of a daily file transfer from many competing E-commerce sites. A daily feed presents static information to the online shopper and cannot accommodate the rapid frequency at which E-commerce sites must react to market changes.
The speed at which a price comparison web site can retrieve and present information is important from the perspective of the user. Consequently, a new breed of efficient price comparison web sites have emerged with the ability to present real-time information to the online shopper. The online shopper, however, needs to be within the confines of these efficient web sites to attain benefit.

By way of contrast, DDGML collects, filters and presents information without the limitations of web site boundaries. The online shopper can interact and discover real-time comparative information from within a participating E-commerce site by utilizing a DDGML Smart Agent. The DDGML Smart Agent resides on an online merchant's product page. Each Smart Agent operates transparent to the user. Each Smart Agent may be customized in the way it feeds real-time information to the new price comparison engine.

A DDGML button resides on a product page as well. This permits the comparison interaction of the online shopper with the price comparison server. Once opted. it serves information to the online shopper.

DDGML utilizes CI messaging. CI messaging streamlines a consumer's E-commerce research to just one mouse click. The online shopper interacts with a DDGML button to instantly compare product information on an E-commerce site with that of other E-commerce sites selling the same product. CI messaging then presents the information that supports which online store is the shopper's best deal.

DDGML also utilizes SDD. SDD monitors predefined research mechanics against a set of target web sites selected by the online merchant. The information SDD collects ranges from simple product pricing to complex online store practices and promotions. It makes use of comparison logic to only report those changes in a target Web site that are relevant, strategic changes. SDD provides an online merchant the ability to instantly react to a competitor's strategy. This permits the online merchant to attempt to negate the identified strategic act and preserve their E-commerce position.

DDGML proposes a standard way to retrieve information from desired web sites. DDGML is an XML application and is based upon a set of meta-tags, which provide a variety of information about a target web sites and the products offered. XML is a variant of HTML and can be used to gather a variety of information from target web sites in real time. This information can include any type of information, such as prices for one or more products, an e-commerce store particulars and shipping costs charged by the store. Examples of such applications could be web sites that provide various comparisons for a given product to a user.

DDGML has a number of benefits as opposed to the traditional methods of parsing HTML. Information retrieved via DDGML is much less as compared to the retrieval of a complete web page, thereby eliminating the need for large bandwidth data transfers. Only the required information is retrieved and no repetitious data elements are downloaded. Since DDGML provides a standard format, it can easily be extended to meet the growing business needs. A standard set of tools can be used to form the required DDGML and standards tools can then be used to extract information from the data.

It is worthy to note that any reference in the specification to "one embodiment" or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the invention. The appearances of the phrase "in one embodiment" in various places in the specification are not necessarily all referring to the same embodiment.

Referring now in detail to the drawings wherein like parts are designated by like reference numerals throughout, there is illustrated in FIG. 1 a system suitable for

practicing one embodiment of the invention. As shown in FIG. 1, a system 100 comprises a user device 102 connected to a price comparison server 104 via a network 106. In this example, network 106 utilizes the hardware and software necessary to transfer information in accordance with Internet technology such as the Transmission Control Protocol (TCP) and Internet Protocol (IP), and related technology such as the World Wide Web (WWW) programming languages and browser technology (e.g., Internet Explorer by Microsoft Corporation). Price Information Servers 108, 110 and 112 (i.e., 1, 2 and N) are also connected to the network 106. In this embodiment of the invention, user device 102 is a personal computer (PC), equipped with a input device (e.g., keyboard and/or mouse), an output device (e.g., a monitor), web browser software (e.g., Internet Explorer 5.0), and a network interface capable of communicating information over network 106 (e.g., network interface card, modem, etc.).

FIG. 2 is a block diagram of a price comparison server in accordance with one embodiment of the invention. FIG. 2 shows a price comparison server 200 comprising a processor 202, a memory 204, and a bus adapter 214, each of which is connected to a processor/memory bus 216 and an Input/Output (I/O) bus 218 via bus adapter 214. Further, server 200 contains a network interface 222, mass storage device 224, monitor 226, keyboard 228 and database 230, each of which is connected to I/O bus 218 via an I/O controller 220.

In one advantageous embodiment of the invention, server 200 is a processor-based PC system. Processor 202 comprises any processor capable of providing the speed and functionality desired for the embodiments of the invention. For example, processor 202 could include the Pentium® family of processors made by Intel Corporation, or the 68XXX family of processors made by Motorola. Further, monitor 226 may be any means for displaying analog signals, such as a VGA monitor. I/O controllers 220 may be any means for controlling the flow of information between I/O bus 218 and the various I/O devices such as network interface 222, mass storage device 224, monitor 226, keyboard 228 and database 230. Bus adapter 214 may be any means suitable for transferring data back and forth between processor/memory bus 216 and I/O bus 218.

For the purpose of this application, memory 204 and storage device 224 are machine readable mediums and could include any medium capable of storing instructions adapted to be executed by a processor. Some examples of such media include, but are not limited to, read-only memory (ROM), random-access memory (RAM), programmable ROM (PROM), erasable programmable ROM (EPROM), electronically erasable programmable ROM (EEPROM), dynamic RAM (DRAM), flash memory, magnetic disk (e.g., floppy disk and hard drive), optical disk (e.g., CD-ROM), and any other device that can store digital information. In one embodiment, the instructions are stored on the medium in a compressed and/or encrypted format. As used herein, the phrase "adapted to be executed by a processor" is meant to encompass instructions stored in a compressed and/or encrypted format, as well as instructions that have to be compiled or installed by an installer before being executed by the processor. Further, server 200 may contain various combinations of machine readable storage devices through other I/O controllers, which are accessible by processor 202 and which are capable of storing a combination of computer program instructions and data.

Network interface 220 may be any suitable means for controlling communication signals between network devices using a desired set of communications protocols, services and operating procedures. In this embodiment of the invention, network interface 216 operates in accordance with the TCP/IP protocols. Network interface 216 also includes connectors for connecting interface 216 with a suitable communications medium. Those skilled in the art will understand that network interface 216 may receive communication signals over any suitable medium such as twisted-pair wire, co-axial cable, fiber optics, radio-frequencies, and so forth.

Memory 204 and device 224 store instructions that when executed by a processor (e.g., processor 202) performs the functionality for the various embodiments of the invention. In one embodiment of the invention, the instructions are separated into three modules, that is, a price comparison module 206, a price update module 208, and a DDGML module 210. It can be appreciated, however, that the functions performed by these modules can be further separated into more modules, combined together to form one module, or be distributed throughout the system, and still fall within the scope of the

invention. Further, although this embodiment of the invention implements the functionality of these modules in software, it can be appreciated that the functionality of these modules may be implemented in hardware, software, or a combination of hardware and software, using well-known signal processing techniques.

Price comparison module 206 performs the SDD function for the price comparison server. Price comparison module 206 generates an HTML document having a price comparison user interface for a user to gather information regarding current pricing information for a desired product or service. The price comparison user interface can be designed to allow the input of pricing parameters desirable to the user, such as a list of prices and companies, ordering of the list, display of a price average, and so forth. Price comparison module 206 receives, compares, organizes and displays the pricing information in accordance with the desired parameters.

Price update module 208 performs the CI messaging for the price comparison server. Price update module 208 receives the pricing parameters and generates a pricing query in accordance with such parameters. Price update module 208 then sends the pricing query to a price information server (e.g., price information servers 1-N) via the user's web browser.

DDGML Module 210 provides all the components necessary to generate and execute the DDGML instructions. For example, since DDGML is a variant of XML, an XML program would reside on server 200 as part of DDGML module 210. Furthermore, a COM object program (e.g., ddgml.dll) written using, for example, Visual Basic (VB) 6.0 (requiring VB 6.0 run-time mode) would also comprise a part of DDGML module 210. Other DDGML executable files and data files could be added in accordance with the desired functionality of the online merchant or user needs.

FIG. 3 is a block diagram of a price information server in accordance with one embodiment of the invention. FIG. 3 shows a price information server 300 that is similar in design to server 200 described with reference to FIG. 2. Thus, elements 302, 304, 308, 310, 314, 316, 318, 320, 322, 324, 326, 328 and 330 of server 300 are similar in structure and function as corresponding elements 202, 204, 208, 210, 214, 216, 218, 220, 224, 226, 228 and 230 of server 200. Memory 304 of server 300, however, also contains a

smart agent module 306. Although memory 304 does not store a computer program module similar to price comparison module 206, it can be appreciated that this functionality can be implemented as part of server 300 and still fall within the scope of the invention, as discussed in more detail below.

Smart agent module 306 performs the function of receiving the pricing query from price comparison server 104. Smart agent module 306 retrieves the pricing information in accordance with the pricing query, generates a response in a DDGML format, and sends the response to price comparison server 104.

The operation of system 100, server 200 and server 300 will be described in more detail with reference to FIGS. 4 and 5. Although each flow chart includes a particular sequence of steps, it can be appreciated that the sequence of steps merely provides an example of how the general functionality described herein can be implemented. Each sequence of steps do not have to be executed in the order presented unless otherwise indicated.

FIG. 4 is a block flow diagram of the steps performed by a price comparison server in accordance with one embodiment of the invention. As shown in FIG. 4, a request for pricing information is received at a first network device at step 402. An example of a first network device includes price comparison server 200. The request could come, for example, from a user seeking current pricing information or an online merchant wanting to determine what the competition is doing. A pricing query is generated for at least a second network device in HTTP format at step 404. An example of a second network device(s) includes price information servers 1-N. The pricing query is sent to the at least one second network device at step 406. Pricing information in response to the pricing query is received at step 408. If there is more than one set of pricing information at step 410, a comparison of the multiple sets of pricing information is performed at step 412. If there is only one set of pricing information, the pricing information is displayed at step 414.

The steps performed by, for example, price comparison server 200 and as shown in FIG. 4 may be better understood by way of example. Price comparison server 200 can update pricing information quicker than conventional technologies due to the use of

DDGML. DDGML includes a set of meta tags used to exchange information, such as price, product and store information, between two or more web sites. In this embodiment of the invention, all data is transmitted using HTTP and can be made secure if necessary. Implementation of DDGML requires some modification on part of the target web site. Ready made tools can be used to enable a site to use DDGML.

By way of example, it is assumed that a web site exists that permits users to perform price comparisons for any number of products or services. The user initiates a connection with the Internet via his/her home computer. The user types in a URL for a price comparison web site, such as "www.pricecomparison.com," in his/her web browsing software. An HTML document is displayed on the users computer monitor, which has among other things, a hypertext link to the price comparison feature for one or more products and/or services. The user uses the mouse to click on the hypertext link, and the price comparison module 206 begins the price comparison process.

The user's actions sends a request for pricing information to price comparison module 206 of price comparison server 200. Price comparison module 206 sends the request to price update module 208. Price update module 208 generates a pricing query in accordance with certain pricing parameters generated by the user or preset by the designers of the web site. The pricing query is sent to price information server 300 via an HTTP request, using the conventional FORM GET method, and in accordance with the DDGML format. An example of the DDGML request sent to the target web site comprises the following attributes:

- storeInfo=Y|N
- prodInfo=prodKey{|prodKey}
- promoInfo=Y|N

An example of such a request is:

"http://www.pcwonders.com/getPrice.asp?storeInfo=Y&prodInfo=1234|8040u&promoInfo=N"

The above URL is designed to retrieve the store information for a web site referred to as "pcwonders," product information for products numbers 1234 and 8040u and no promotional information would be retrieved.

Once the web site "pcwonders" receives and processes the request for pricing information, it will send back the pricing information to price comparison server 200. Once the information is returned to price comparison server 200, price update module 208 parses the pricing information using a DDGML parser that is part of DDGML module 210. In one embodiment of the invention, the DDGML parser is a standard XML parser, such as the one which comes with Internet Explorer 5.0 made by Microsoft Corporation, or a Java based third party parser could be used. If there is more than one set of pricing information, price comparison module 204 compares the multiple sets of pricing information. The parsed data could then be used to create the required HTML pages and displayed to the user using conventional techniques.

FIG. 5 is a block flow diagram of the steps performed by a price information server in accordance with one embodiment of the invention. As shown in FIG. 5, the pricing query sent by the first network device is received at the second network device at step 502. The second network device sends the request to a pricing interface program at step 504. An example of a pricing interface program is smart agent module 304. The pricing interface program parses the DDGML pricing query sent via HTTP into separate pricing data at step 506. A data base search request is sent from the pricing interface program to a data base interface program at step 508. Pricing information corresponding to the separate pricing data is retrieved from the database at step 510. The pricing information is received at the pricing interface program at step 512. The received pricing information is converted to DDGML format at step 514, and is sent to the first network device via HTTP at step 516.

The steps performed by, for example, price information server 300 and as shown in FIG. 5 may be better understood by way of the previous example given with respect to FIG. 4. The pricing query sent from price comparison server 200 is received by smart agent module 304 of price information server 300. Smart agent module 304 could be, by way of example, an Active Server Page (ASP), a Common Gateway

Interface (CGI) script or an ISAPI/NSAPI [PLEASE DEFINE] extension DLL. The purpose of this page is to decipher the DDGML pricing query, which specifies what information is requested. This information includes items such as, for example, store particulars, product-pricing information, and so forth. A single request can be used to retrieve prices for multiple products. The intercepting page then retrieves the required information from database 330 of price information server 300, converts the required information into DDGML, and sends it back via HTTP. A COM object can be used to create the required DDGML (if the web site is running on an NT box) or a Practical Extension and Reporting Language (PERL) script could be used if UNIX is the host operating system. Standard implementations of these components and scripts can be provided to interested web sites.

An example of a pricing query is composed of three groups of information:

- Store information
- Product information
- Promotional information

Store information comprises of various properties, which pertain to an on-line store. These properties consist of the store name: store URL, store address, contact person name, contact person phone number and type of business this store is in. The product information consists of pricing, availability, shipping costs for the given product, a URL that can be used to add this product to a shopping cart along with a URL that can be used to navigate to the product detail page for that specific product. Promotional information consists of any special promotional information at the store level or at the product level. This promotional could be things like free shipping for all products or for a single product, promotions such as buy one get one free etc.

DDGML is a variant of XML and comprises a set of meta-tags for use in specifically identifying, formatting and sending pricing information. Following is a

partial list of tags supported by DDGML and what those tags can be used for:

Element Name	Description	Type	#	Default Value
DDGMLResults	Marks the beginning of a query.	-	1	
StoreInfo	Beginning of store information.	-	0 or 1	
StoreName	Name of the store.	String	0 or 1	
StoreURL	URL to navigate to the store=s main page.	String	0 or 1	
StoreAddress	Beginning of address for the store.	-	0 or 1	
street1	First line for street address.	String	1	
street2	Second line for street address.	String	0 or 1	
street3	Third line for street address.	String	0 or 1	
State	State information.	String	1	
Zip	Zip code.	String	1	
contactPerson	Contact person=s name.	String	0 or 1	
contactPhone	Contact=s phone number.	String	0 or 1	
ProductInfo	Marks the beginning of product information.	-	0 or 1	
Product	Beginning of single product information.	-	1 or more	
Name	Name of the product.	String	0 or 1	
Price	Price of the product.	Numeric	1	
PartNumber	Part number of the product.	String	0 or 1	
Availability	Availability flag.	String	1	Y
Quantity	Quantity on hand.	Numeric	0 or 1	

Element Name	Description	Type	#	Default Value
SalesTax	Sales tax if applicable	Numeric	1	
CartURL	URL to add this item to the shopping cart.	String	0 or 1	
Shipping	Beginning of any shipping rates information.	-	0 or 1	
shippingAmount	Shipping amount for this product.	Numeric	1	0.0
Method	Attribute for shippingAmount.	String	1	
promoInfo	Any applicable promotional information for this product.	String	0 or 1	A@

The following is an example of a sample DDGML request which would be generated in response to the following request:

http://www.pcwonders.com/getPrices.asp?storeInfo=Y&prodInfo=80400u&promoInfo=Y

```
<?xml version="1.0"?>
<!DOCTYPE DDGMLResults SYSTEM "ddgml.dtd">
<DDGMLResults>
    <storeInfo>
        <storeName>pcWonders</storeName>
        <storeURL>http://www.pcwonders.com</storeURL>
        <storeAddress>
            <street1>9 Whippany Road</street1>
            <street2>Suite 12</street2>
            <state>NJ</state>
            <zip>07981</zip>
        </storeAddress>
        <contactPerson>John Do</contactPerson>
```

```
            <contactPhone>(973) 555-1212</contactPhone>
        </storeInfo>

        <productInfo>
            <product>
                <name>Palm V</name>
                <partNumber>80400u</partNumber>
                <quantity>1000</quantity>
                <shipping>
                    <shippingAmount method="UG">10.04</shippingAmount>
                    <shippingAmount method="U2">20.04</shippingAmount>
                </shipping>
                <price>108.35</price>
                <salesTax>0</salesTax>
                <availability>Y</availability>
            <cartURL>http://www.pcwonders.com/addToCart.asp?pID=80400u</cartURL>
            </product>
        </productInfo>

        <promoInfo>
            Buy one get one free.
        </promoInfo>
</DDGMLResults>
```

The following is the Document Type Definition for DDGML:

```
<!--DDGML DTDB>
<!ELEMENT DDGMLResults          ((storeInfo)?, (productInfo)?, (promoInfo)?)>

<!ELEMENT storeInfo             ((storeName, storeURL, storeAddress)?,
                                (contactPerson)?, (contactPhone)?)>
```

```
<!ELEMENT storeName          (#PCDATA)>
<!ELEMENT storeURL           (#PCDATA)>
<!ELEMENT contactPerson      (#PCDATA)>
<!ELEMENT contactPhone       (#PCDATA)>

<!ELEMENT storeAddress       (street1, (street2)?, (street3)?, state, zip)>
<!ELEMENT street1            (#PCDATA)>
<!ELEMENT street2            (#PCDATA)>
<!ELEMENT street3            (#PCDATA)>
<!ELEMENT state              (#PCDATA)>
<!ELEMENT zip                (#PCDATA)>

<!ELEMENT productInfo        (product)+>
<!ELEMENT product            ((name)?, (partNumber)?, (quantity)?, (shipping)?,
                              price, availability, (cartURL)?))>
<!ELEMENT name               (#PCDATA)>
<!ELEMENT partNumber         (#PCDATA)>
<!ELEMENT quantity           (#PCDATA)>
<!ELEMENT price              (#PCDATA)>
<!ELEMENT availability       (#PCDATA)>
<!ELEMENT cartURL                  (#PCDATA)>

<!ELEMENT shipping           (shippingAmount)+>
<!ELEMENT shippingAmount     (#PCDATA)>
<!ATTLIST shippingAmount method CDATA      #REQUIRED>

<!ELEMENT promoInfo          (#PCDATA)>
```

The following outlines the components necessary to integrate DDGML into an ASP based web site. These components are made part of the DDGML module 210

and/or 310 used by price comparison server 200 and price information server 300, respectively. These components are:

- An Active Server Page/CGI script that serves the results
- The DDGML COM object

The ASP acts as the interface between the web site that is requesting the price comparison, and the web site that is providing the comparison data. A call is made to this ASP from the requesting web site. This call is a URL with parameters appended to its end (performing a GET method). In one embodiment of the invention, the name of the Active Server Page is AddgmlServer.asp@ and is located on the web site under a virtual directory accessible from the web root, called DDGML.

The request sent to the target web site is made up of the following exemplary attributes:

- storeInfo (Y or N)
- prodInfo (contains a list of product numbers delimited by the | character)
- promoInfo (Y or N)
- catInfo (category of product we are searching for, e.g., dvd, vid, spo, ele, com, bok, cd, toy, msc)

The *storeInfo* parameter dictates if the request for any store information is required or not. The store information consists of store address, contact information etc. The possible value for *storeInfo* is Y (store information is required) or N (store information is not required). The *prodInfo* parameter lists a single or multiple product keys for which the price information is required. If no product information is required then this parameter could be omitted. The *promoInfo* parameter is used to gather any promotional information for the target store. The possible values *promoInfo* can have are Y (promotional information is required) or N (promotional information is not required). The *catInfo* parameter defines the type of category to which this information belongs. Stores which sell more than one line of products can use this parameter to optimize their search. This parameter dictates what kind of part is being passed in the *prodInfo* parameter. In one embodiment of the invention, this parameter has the following values:

dvd DVD part number

vid VHS video

spo Sports items

ele Electronic goods

com Computer related product

bok Book

cd A music CD

toy Toys

msc Miscellaneous item

Once the pricing query is parsed, the ASP instantiates the DDGML COM object. The COM object contains the necessary object model to generate the required DDGML. In one embodiment of the invention, the DDGML COM object supports the following interface:

Document object (CDocument)		
UseDTD	Property	Boolean B Whether to include the DTD link or not.
DTD	Property	String B The link to a Document Type Definition file.
ProductInfo	Property (read-only)	CProducts B Returns a reference to the Products collection.
PromoInfo	Property (read-only)	CPromotion B Returns a reference to the promotion information object.
StoreInfo	Property (read-only)	CStore B Returns a reference to the store information object.
DDGML	Method	Returns a string containing the DDGML. The parameter bRefresh decides if the DDGML should be refreshed or DDGML from the previous call should be returned.

		Prototype: Public Function DDGML([ByVal bRefresh As Boolean = True]) As String

Products object (CProducts)		
Count	Property	Long B Total number of products in the collection (usually this will either be 0 or 1).
Item	Property	CProduct B Returns a single product item.
Add	Method	Adds a product to the collection. It returns the newly added product. Prototype: *Add(ByVal PartNumber As String, [ByVal ProductName As String], [ByVal Quantity As Long], [ByVal Price As Double], [ByVal SalesTax As Double], [ByVal Availability As Boolean], [ByVal CartURL As String], [ByVal ProductPageURL As String], [ByVal sKey As String]) As CProduct*
Remove	Method	Removes a product from the collection. The method takes one parameter which could either be an index. or a key.
DDGML	Method	Returns a string containing the DDGML. The parameter bRefresh decides if the DDGML should be refreshed or DDGML from the previous call should be returned. Prototype: Public Function DDGML([ByVal bRefresh As Boolean = True]) As String
Product object (CProduct)		

Availability	Property	Boolean B If the product is available or not.
CartURL	Property	String B URL to add the product to the shopping cart.
PartNumber	Property	String B Part number of the product.
Price	Property	Double B Price of the product.
ProductName	Property	String B Name of the product.
ProductPageURL	Property	String B URL to the product page.
Quantity	Property	Long B Quantity on hand.
SalesTax	Property	Double B Sales tax if any.
ShippingRates	Property (read-only)	CShippingRates B Returns a reference to the shipping rates collection.
DDGML	Method	Returns a string containing the DDGML. The parameter bRefresh decides if the DDGML should be refreshed or DDGML from the previous call should be returned. Prototype: Public Function DDGML([ByVal bRefresh As Boolean = True]) As String
CshippingRates		
Count	Property	Long B Total number of products in the collection (usually this will either be 0 or 1).
Item	Property	CShippingRate B Returns a single shipping rate item.
Add	Method	Adds a shipping rate information to the collection. It returns the newly added shipping method. Prototype: *Public Function Add(ByVal ShippingAmount As Double, ByVal ShippingMethod As*

		DDGMLShippingMethods, [ByVal sKey As String]) As CShippingRate
Remove	Method	Removes a shipping rate from the collection.
DDGML	Method	Returns a string containing the DDGML. The parameter bRefresh decides if the DDGML should be refreshed or DDGML from the previous call should be returned. Prototype: Public Function DDGML([ByVal bRefresh As Boolean = True]) As String
CshippingRate		
ShippingAmount	Property	Double B The shipping amount.
ShippingMethod	Property	Enum B Enumerated type for shipping methods.
ShippingMethodName	Property	String B Code of the shipping method.
GetShippingMethodName	Method	Returns the description for the shipping method.
DDGML	Method	Returns a string containing the DDGML. The parameter bRefresh decides if the DDGML should be refreshed or DDGML from the previous call should be returned. Prototype: Public Function DDGML([ByVal bRefresh As Boolean = True]) As String
Cstore		
ContactPerson	Property	String B Name of the contact person at the store.
ContactPhone	Property	String B Phone number of the contact.

State	Property	String B State (part of address).
StoreName	Property	String B Name of the store.
StoreURL	Property	String B A URL pointing to the store.
Street1	Property	String B Address line 1.
Street2	Property	String - Address line 2.
Street3	Property	String - Address line 3.
Zip	Property	String B Zip code.
DDGML	Method	Returns a string containing the DDGML. The parameter bRefresh decides if the DDGML should be refreshed or DDGML from the previous call should be returned. Prototype: Public Function DDGML([ByVal bRefresh As Boolean = True]) As String

The ASP sets values to the above properties after instantiating the ACDocument class@ in the DDGML COM object.

In one embodiment of the invention, DDGML modules 210 and/or 310 utilizes the following files to implement DDGML for a web site:

ddgml.inc	an include file which contains VBScript functions to parse the QueryString.
ddgml.dll	COM object (requires VB 6.0 run-time).
ddgmlServer.asp	A sample ASP file (this file would require customization). This is the file where your web site specific code to search would be placed.

Although various embodiments are specifically illustrated and described herein, it will be appreciated that modifications and variations of the present invention are covered by the above teachings and within the purview of the appended claims without departing from the spirit and intended scope of the invention. For example, although a price comparison module is not shown for price information server 300, it can be appreciated that such a module could be included to perform comparison functions for, for example, an online merchant.

What is claimed is:

1. A method for updating pricing information over a network, comprising:
receiving a request for pricing information at a first network device;
sending said request to a second network device; and
receiving a first set of pricing information in a dynamic data gathering markup language (DDGML) format.

2. The method of claim 1, wherein said sending said request comprises:
generating a pricing query to said second network device in a hypertext transfer protocol (HTTP) format; and
sending said pricing query to said second network device.

3. The method of claim 2, wherein said pricing information comprises separate pricing data, and said DDGML format comprises encapsulating said separate pricing data within a set of predefined tags.

4. The method of claim 3, wherein said set of predefined tags are meta tags defined using an Extensible Markup Language (XML).

5. The method of claim 1, further comprising:
converting said first set of pricing information in said DDGML format to a second format; and
displaying said converted first set of pricing information in said second format.

6. The method of claim 5, wherein said second format is a hypertext markup language (HTML) format.

7. The method of claim 6, wherein said converting comprises:
sending said first set of pricing information to an XML parser:

parsing said first set of pricing information to a set of interim data; and
generating an HTML document based on said interim data.

8. The method of claim 1, wherein said pricing information comprises at least one attribute from a group of attributes comprising store information, product information and promotional information.

9. The method of claim 8, wherein said store information comprises at least one attribute from a group of attributes comprising a store name, store user resource locator (URL), physical address, contact person and contact information.

10. The method of claim 8, wherein said product information comprises at least one attribute from a group of attributes comprising a product name, product price, part number, availability indicator, quantity indicator, sales tax, shopping cart user resource locator (URL), shipping rate, shipping amount, and shipping method.

11. The method of claim 8, wherein said promotional information comprises at least one attribute from a group of attributes comprising discounts, rebates, gifts and special offers.

12. The method of claim 1, further comprising
sending said request to a third network device;
receiving a second set of pricing information in said DDGML format;
comparing said first set with said second set; and
displaying said pricing information in accordance with said comparison.

13. The method of claim 1, further comprising:
receiving said request at said second network device:
retrieving said first set of pricing information; and

sending said first set of pricing information to said first network device.

14. The method of claim 13, wherein said retrieving comprises:

sending said request to a pricing interface program;

parsing said request to separate pricing data;

sending a data base search request from said pricing interface program to a data base interface program;

retrieving said separate pricing data from said data base;

receiving said separate pricing data at said pricing interface program; and

converting said separate pricing data to said DDGML format.

15. The method of claim 14, wherein said pricing interface program is a common gateway interface (CGI) application program.

16. The method of claim 15, wherein said CGI application program comprises a set of computer program segments coded from at least one computer language from a group of computer languages comprising C, C++, Pascal, AppleScript, Visual Basic and Practical Extraction and Reporting Language (PERL).

17. The method of claim 14, wherein said pricing interface program is an active server page (ASP).

18. The method of claim 14, wherein said pricing interface program is an ISAPI/NSAPI extension DLL.

19. A machine-readable medium whose contents cause a computer system to perform price updates over a network, by performing the steps of:

receiving a request for pricing information at a first network device;

sending said request to a second network device; and

receiving a first set of pricing information in a dynamic data gathering markup

language (DDGML) format.

20. The machine-readable medium of claim 19, wherein said sending said request comprises:

generating a pricing query to said second network device in a hypertext transfer protocol (HTTP) format; and

sending said pricing query to said second network device.

21. The machine-readable medium of claim 20, wherein said pricing information comprises separate pricing data, and said DDGML format comprises encapsulating said separate pricing data within a set of predefined tags.

22. The machine-readable medium of claim 21, wherein said set of predefined tags are meta tags defined using an Extensible Markup Language (XML).

23. The machine-readable medium of claim 19, further comprising:

converting said first set of pricing information in said DDGML format to a second format; and

displaying said converted first set of pricing information in said second format.

24. The machine-readable medium of claim 23, wherein said second format is a hypertext markup language (HTML) format.

25. The machine-readable medium of claim 24, wherein said converting comprises:

sending said first set of pricing information to an XML parser;

parsing said first set of pricing information to a set of interim data; and

generating an HTML document based on said interim data.

26. The machine-readable medium of claim 19, wherein said pricing information comprises at least one attribute from a group of attributes comprising store

information, product information and promotional information.

27. The machine-readable medium of claim 26, wherein said store information comprises at least one attribute from a group of attributes comprising a store name, store user resource locator (URL), physical address, contact person and contact information.

28. The machine-readable medium of claim 26, wherein said product information comprises at least one attribute from a group of attributes comprising a product name, product price, part number, availability indicator, quantity indicator, sales tax, shopping cart user resource locator (URL), shipping rate, shipping amount, and shipping method.

29. The machine-readable medium of claim 26, wherein said promotional information comprises at least one attribute from a group of attributes comprising discounts, rebates, gifts and special offers.

30. The machine-readable medium of claim 19, further comprising
sending said request to a third network device;
receiving a second set of pricing information in said DDGML format;
comparing said first set with said second set; and
displaying said pricing information in accordance with said comparison.

31. The machine-readable medium of claim 19, further comprising:
receiving said request at said second network device;
retrieving said first set of pricing information; and
sending said first set of pricing information to said first network device.

32. The machine-readable medium of claim 31, wherein said retrieving comprises:
sending said request to a pricing interface program;

parsing said request to separate pricing data;

sending a data base search request from said pricing interface program to a data base interface program;

retrieving said separate pricing data from said data base;

receiving said separate pricing data at said pricing interface program; and

converting said separate pricing data to said DDGML format.

33. The machine-readable medium of claim 32, wherein said pricing interface program is a common gateway interface (CGI) application program.

34. The machine-readable medium of claim 33, wherein said CGI application program comprises a set of computer program segments coded from at least one computer language from a group of computer languages comprising C, C++, Pascal, AppleScript, Visual Basic and Practical Extraction and Reporting Language (PERL).

35. The machine-readable medium of claim 33, wherein said pricing interface program is an active server page (ASP).

36. The machine-readable medium of claim 33, wherein said pricing interface program is an ISAPI/NSAPI extension DLL.

37. An apparatus to update pricing information over a network, comprising:

a price comparison module;

a price update module connected to said price comparison module; and

a dynamic data gathering markup language module connected to said price comparison module and said price update module.

38. An apparatus to update pricing information over a network, comprising:

a smart agent module;

a price update module connected to said smart agent module; and

a dynamic data gathering markup language module connected to said smart agent module and said price update module.

39. A system, comprising:

a first server;

a second server; and

a network to transfer information between said first and second server in a dynamic data gathering markup language format.

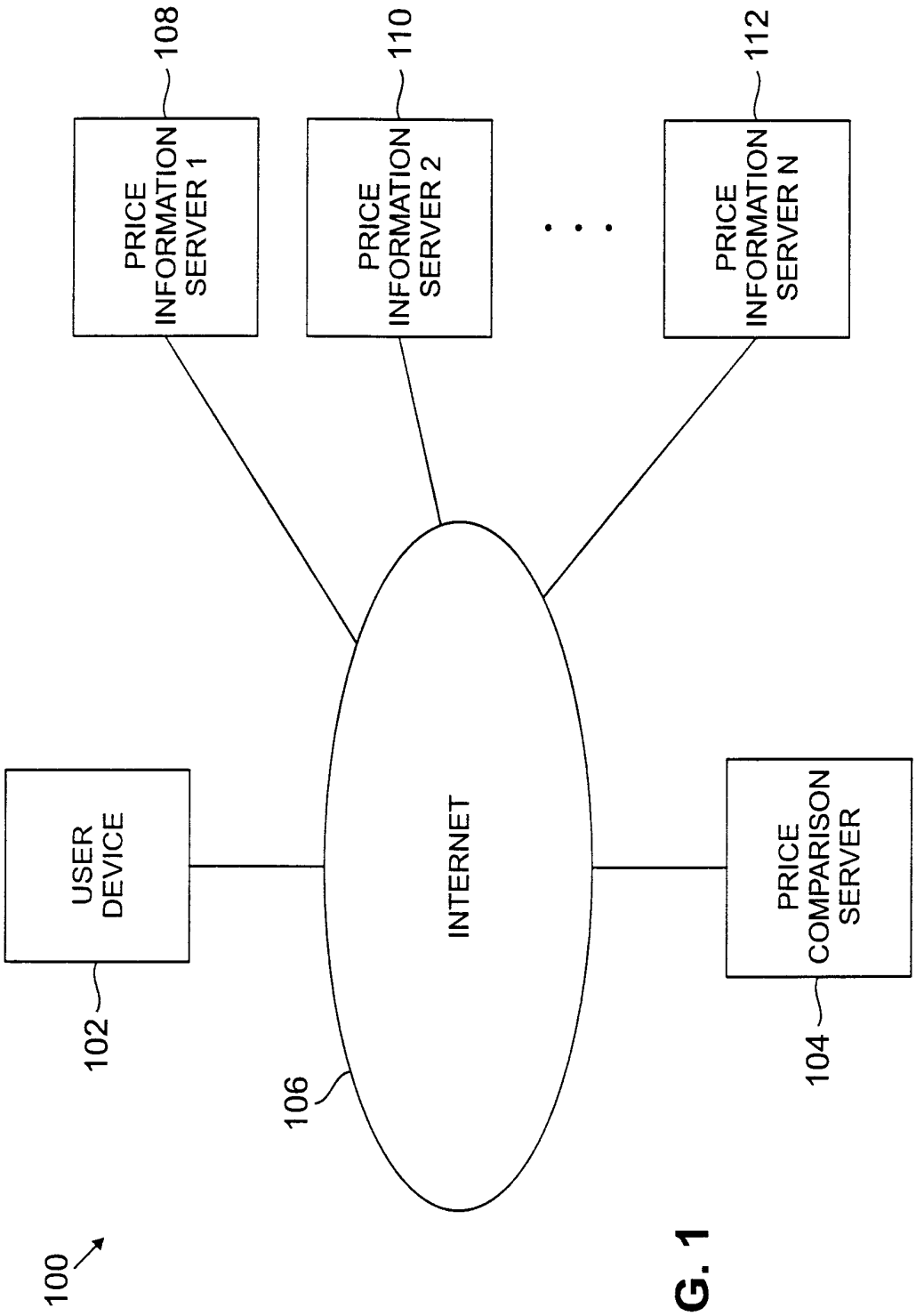

PRICE INFORMATION SERVER 1
108
PRICE INFORMATION SERVER 2
110
PRICE INFORMATION SERVER N
112
USER DEVICE
102
INTERNET
106
PRICE COMPARISON SERVER
104
100

FIG. 1

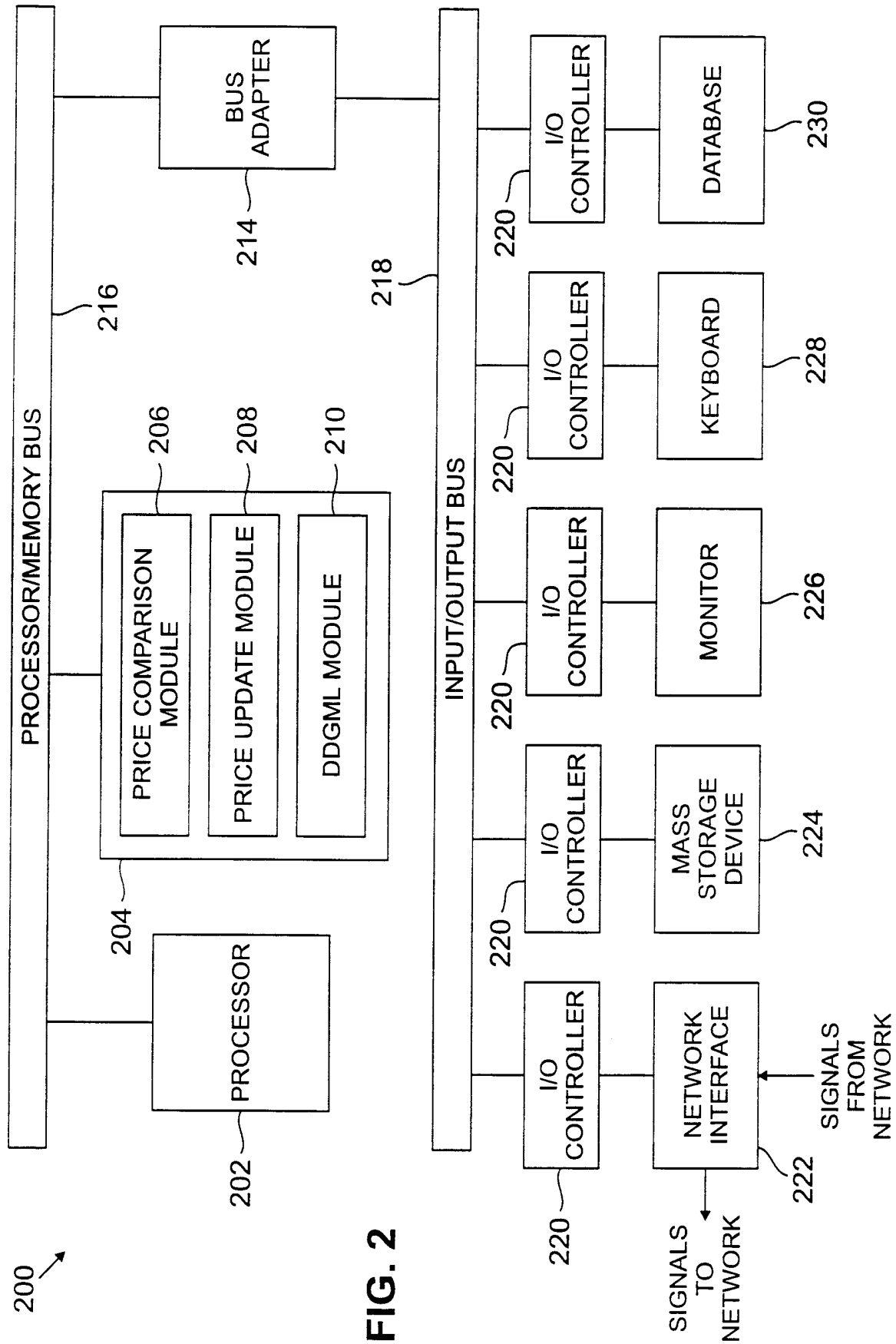

200
PROCESSOR/MEMORY BUS
216
BUS ADAPTER
214
PROCESSOR
202
204
PRICE COMPARISON MODULE
206
PRICE UPDATE MODULE
208
DDGML MODULE
210
218
INPUT/OUTPUT BUS
I/O CONTROLLER
220
DATABASE
230
KEYBOARD
228
MONITOR
226
MASS STORAGE DEVICE
224
NETWORK INTERFACE
222
SIGNALS FROM NETWORK
SIGNALS TO NETWORK

FIG. 2

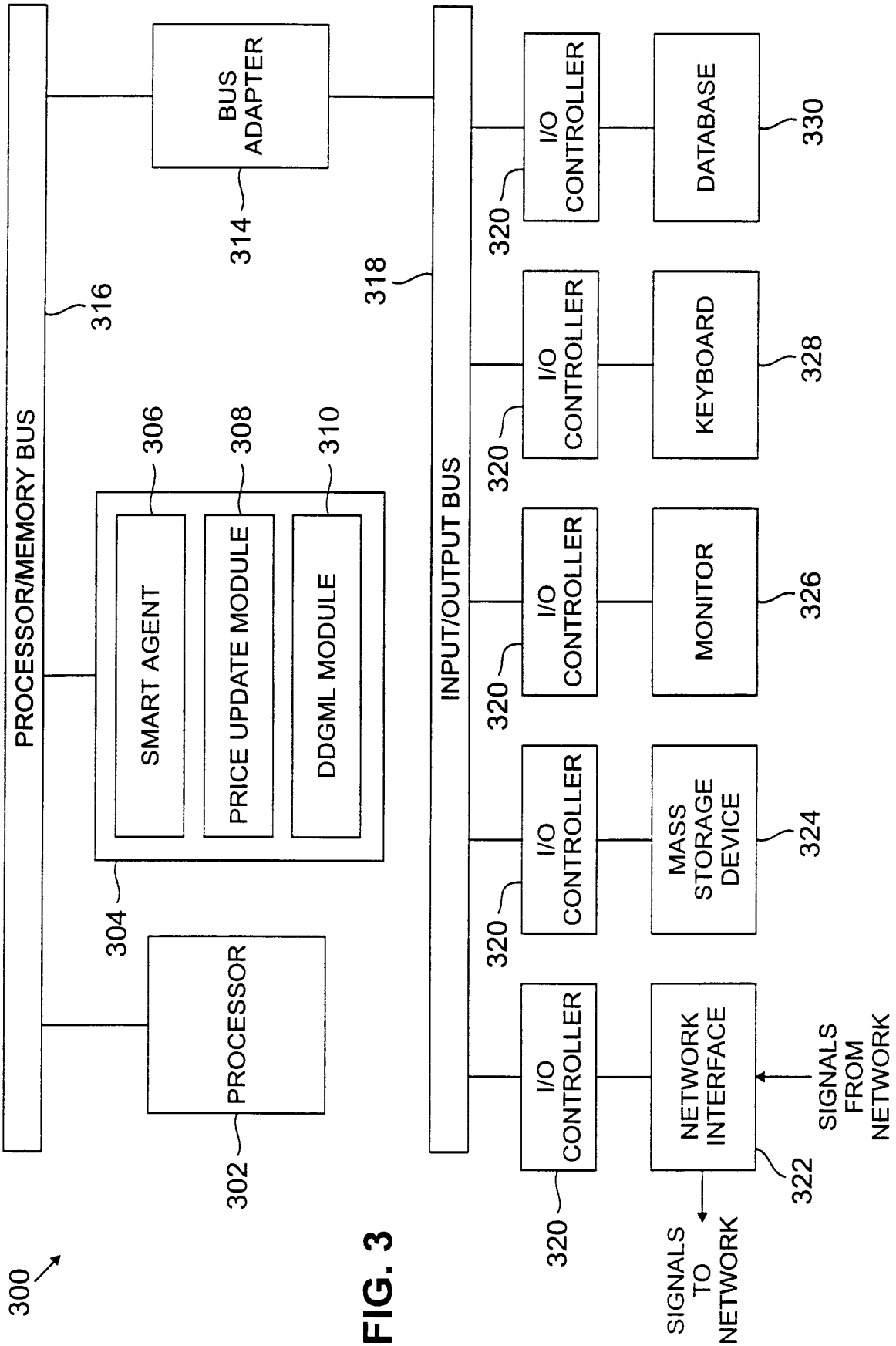

300
PROCESSOR/MEMORY BUS
316
BUS ADAPTER
314
318
INPUT/OUTPUT BUS
PROCESSOR
302
304
SMART AGENT
306
PRICE UPDATE MODULE
308
DDGML MODULE
310
I/O CONTROLLER
320
DATABASE
330
KEYBOARD
328
MONITOR
326
MASS STORAGE DEVICE
324
NETWORK INTERFACE
322
SIGNALS FROM NETWORK
SIGNALS TO NETWORK

FIG. 3

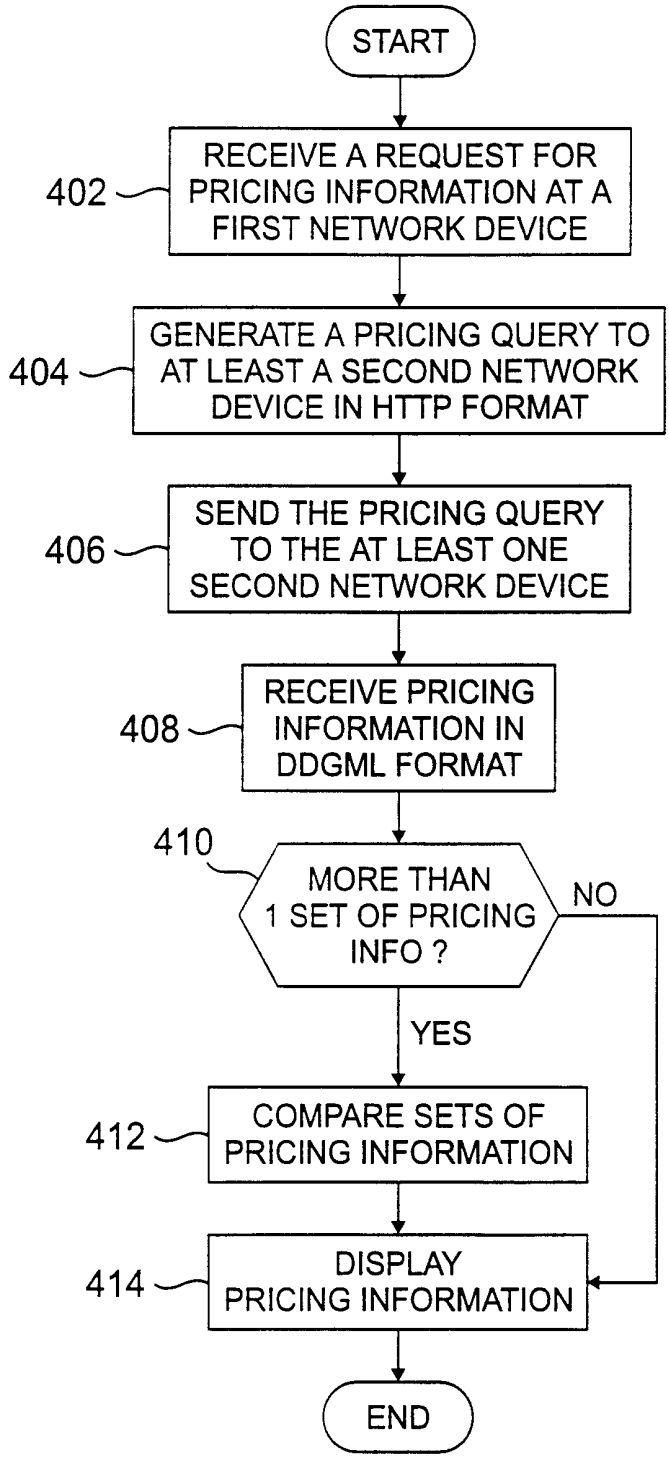

START
RECEIVE A REQUEST FOR PRICING INFORMATION AT A FIRST NETWORK DEVICE
402
GENERATE A PRICING QUERY TO AT LEAST A SECOND NETWORK DEVICE IN HTTP FORMAT
404
SEND THE PRICING QUERY TO THE AT LEAST ONE SECOND NETWORK DEVICE
406
RECEIVE PRICING INFORMATION IN DDGML FORMAT
408
410
MORE THAN 1 SET OF PRICING INFO ?
NO
YES
COMPARE SETS OF PRICING INFORMATION
412
DISPLAY PRICING INFORMATION
414
END

FIG. 4

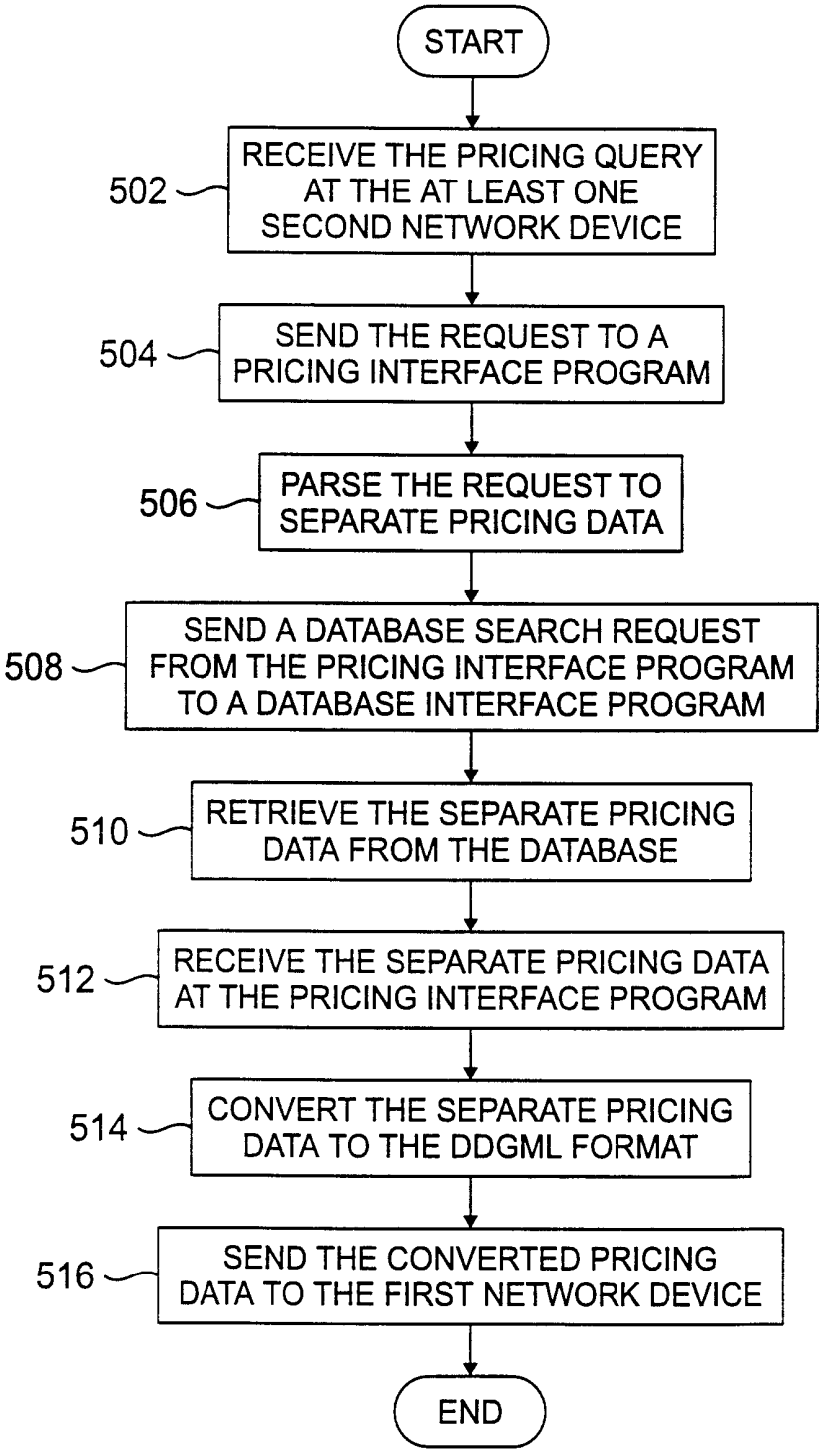

START
502
RECEIVE THE PRICING QUERY AT THE AT LEAST ONE SECOND NETWORK DEVICE
504
SEND THE REQUEST TO A PRICING INTERFACE PROGRAM
506
PARSE THE REQUEST TO SEPARATE PRICING DATA
508
SEND A DATABASE SEARCH REQUEST FROM THE PRICING INTERFACE PROGRAM TO A DATABASE INTERFACE PROGRAM
510
RETRIEVE THE SEPARATE PRICING DATA FROM THE DATABASE
512
RECEIVE THE SEPARATE PRICING DATA AT THE PRICING INTERFACE PROGRAM
514
CONVERT THE SEPARATE PRICING DATA TO THE DDGML FORMAT
516
SEND THE CONVERTED PRICING DATA TO THE FIRST NETWORK DEVICE
END

FIG. 5

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US00/07071

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) :GO6F 17/30

US CL : 707/3, 104, 501, 513; 705/26, 27

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 707/3, 104, 501, 513; 705/26, 27

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

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

Please See Extra Sheet.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5,860,068 A (COOK) 12 January 1999, col. 2, lines 20-67, col. 3, lines 1-47, col. 4, lines 8-67, col. 5, lines 1-67, col. 6, lines 1-67, col. 7, lines 46-67, col. 8, lines 1-67, col. 9, lines 1-25, col. 10, lines 24-67, and col. 11, lines 1-15.	1-39

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

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

Date of the actual completion of the international search 03 JULY 2000	Date of mailing of the international search report 25 JUL 2000
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer KIM VU James R. Matthews Telephone No. (703) 305-4393

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

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US00/07071

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5,897,622 A (BLINN et al) 27 April 1999, col. 1, lines 12-67, col. 2, lines 1-10, col. 3, lines 1-3 and lines 56-67, col. 4, lines 1-26, col. 6, lines 36-67, col. 7, lines 1-67, col. 8, lines 1-67, col. 9, lines 1-43, col. 10, lines 6-67, col. 11, lines 1-67, col. 12, lines 1-67, col. 13, lines 1-13, col. 14, lines 36-61, col. 15, lines 22-48, col. 16, lines 6-67, col. 17, lines 1-67, col. 18, lines 12-24, col. 19, lines 16-17, col. 20, lines 1-36 and lines 53-67, col. 21, lines 1-34, col. 22, lines 28-45, and col. 23, lines 25-61.	1-39
Y	US 5,905,973 A (YONEZAWA et al) 18 May 1999, col. 1, lines 13-67, col. 2, lines 1-67, col. 3, lines 1-6, col. 4, lines 1-67, col. 5, lines 1-67, col. 6, lines 10-67, col. 7, lines 1-10, col. 8, lines 17-64, col. 9, lines 61-67, and col. 10, lines 1-36 and lines 52-64.	1-39

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US00/07071

B. FIELDS SEARCHED
Electronic data bases consulted (Name of data base and where practicable terms used):

WEST
Search terms: network, internet, world wide web, pricing information, data gathering, markup language, XML, query, HTML, parser