A method and apparatus for data retrieval by a computing system and a plurality of agent computers in a distributed network is disclosed. The computing system sends a request to each agent computer to perform a search at a node. The agents perform the searches. The agents then send the resulting data to the computing system for storage in a central database.
Distribute Client Executable

Identify Asset to be Tracked

Determine Query to Run

Has Server Been Blocked from Query?

Delegating Query to Client Executable

Conduct Server Query

Is Server Blocked from Query?

Delegating Query to Client Executable

Receive Queried Data from Client

Prepare Notification For User

FIG. 3
Server Receives Purchased Asset Information

Server Creates Tracking Rule For Purchased Asset

Purchased Asset Active?

Yes

Server Queries Target Website

Is Lower Fair Threshold Reached?

Yes

Report Results To User

Remove Tracking Rule From Tracking Engine

No

Continue

END

FIG. 4
Receive Overlay Rules at Client

Receive Web Page Response at Client

Is Page an Overlay Web Page?

Overlay Customized Content

Render Modified Web Page

Render Received Web Page

END

FIG. 5
Client receives executable code

User executes query at node, code runs in background

Client places user control near data field associated with search results

User clicks on control

Code executes notification to server of target asset and request to track

User creates/accepts/modified tracking rules

Server and/or client produces electronic document

FIG. 7
Distribute Client Executable

Server Receives Purchased Asset Information

Server Creates Tracking Rule For Purchased Asset

Purchased Asset Active?

Remove Tracking Rule Tracking Engine

Server Queries Target Website

Is Server Blocked from Target Websites?

Delegate Query to Client Executable

Is Lower Price Threshold Reached?

Report Results To User

Continue

END
Receive Target Asset Parameters

Receive Characteristic Variance Threshold

Receive Reporting Frequency Setting

Create Entry in Tracking Engine

FIG. 9
SYSTEM AND METHOD FOR RETRIEVING DATA USING AGENTS IN A DISTRIBUTED NETWORK

CROSS-REFERENCE TO RELATED APPLICATION

[0001] The present application for patent claims priority to Provisional Application No. 60/866,433 entitled “System And Method For Tracking Target Assets And Alerting Users Of Changes On A Computer Network,” filed Nov. 20, 2006, attorney docket no. 79789-011, and assigned to the assignee hereof and hereby expressly incorporated by reference herein.

BACKGROUND

[0002] 1. Field

[0003] The present invention relates generally to data retrieval in distributed networks, and more specifically to techniques for retrieving data from nodes using agent machines in the network.

[0004] 2. Description of Related Art

[0005] For a variety of applications, a computing system on a network such as the Internet may be tasked with retrieving data from various locations. One such application is an Internet search engine. Many commercially available search engines engage in the practice of Web scraping to collect data. Web scraping refers to extracting content from websites for the purpose of transforming the content into another format suitable for use in another application. In the case of an Internet search engine, an automated web crawler program may explore the Internet and copy content from millions of websites. The content then can be indexed and made available to users in response to the execution of queries at the search engine website.

[0006] From the standpoint of exposure, Web scraping in the search engine arena often stands to benefit both the sponsor of the scraping program and the website owner. For an established search engine like Google, website owners can benefit greatly from allowing a web crawler to search their content, because it offers the potential to enable many more users to discover and visit their websites than they otherwise would through, for example, happening upon the website during the course of browsing. For this reason, many website owners intentionally place content in a designated area on their websites, with the anticipation that the web crawler will search these areas for content tailored specifically for use in connection with subsequent Internet searches conducted by users.

[0007] For various reasons, many website owners run special blocking code on their websites that attempts to recognize automated scraping programs and prevent them from collecting data on the targeted websites. For example, a company that maintains a travel website for selling airline tickets may elect to limit access to its websites to “human” users—namely, users that are manually running a web browser on a computer and conducting queries in real time at the website over the Internet.

[0008] Many of these blocking programs work by searching for and identifying a node (i.e., a website or other network location) with a particular address that repeatedly executes searches at a target website and returns the search results to the node from the website, often in volume. This may indicate that the identified node is sponsoring a scraping process at the target website. In addition, the blocking programs may explore one or more attributes of the search itself such as, for example, whether repeated queries follow some recognizable pattern or the digital signature left by the node. These and other characteristics of the search often provides clues that the searches are automated, rather than being conducted at the direct behest of a user in real time. In short, where a node querying the target website demonstrates some or all of these characteristics, the blocking program may flag this node as one believed to be running an automated data scraping program. In this event, the blocking program may prevent future access by the node to the target website.

[0009] The businesses and website owners that represent potential targets for web-scraping programs may perceive, in certain instances, that such programs serve to dilute the import or popularity of their websites, to reduce their profitability by giving customers more purchase options from other sources, or to focus consumers on entitlements that do not necessarily benefit the specific objective of the website. As a result they may take measures such as those discussed above to attempt to limit access by certain types of web scraping-type programs, or to exclude such programs altogether from accessing the target website.

[0010] For these types of traditional blocking programs, it is generally important to the website owner that any candidate blocking program considered for use at the target website does not inadvertently prevent what they perceive to be “legitimate” users of client machines from having substantially unhindered access to information at the target website. These legitimate users may, for example, be individuals executing routine queries in a manner intended by the website for topics, products, items or assets, for purchase or otherwise.

[0011] To curtail the inadvertent blocking of the website’s target audience of potential customers, many blocking programs are configured to issue block orders only to those nodes whose activity at the target website satisfies a condition. Such conditions may include, for example, the node’s frequency of visiting a target website, the amount of the target website’s resources used by the node, or the volume of information obtained by the node from the website. Only one or more of these conditions exceed some predetermined threshold would the node be blocked from access. This approach represents a traditional attempt by the website owner to balance the owner’s interest in preventing access to the website by unwanted scraping nodes on one hand, and preserving to the target websites the right of entry for “desirable” users on the other.

[0012] One problem with this conventional approach is that otherwise legitimate data collecting programs may simply be blocked wholesale by e-commerce based business and other websites, without regard to the numerous advantages that sponsors of these programs may offer to a variety of classes of individuals. From a legal standpoint, the objectives of the entity owning a particular web scraping program may be entirely legitimate. Such scraping programs may in actuality result in the provision of necessary or useful services and benefits to the business owner, the relevant consumer class, or both. This is particularly true where the data blocked from access constitutes government-published data, or data types involving minimal or no restrictions of use.

[0013] In the above example of the travel website, a consumer may wish to purchase an airline ticket on the Internet. To get the lowest possible price of a ticket, the consumer may well be required to spend a considerable amount of time...
visiting a plethora of websites, such as some of the major travel websites as well as the airlines’ own websites. If, however, a data retrieval program performs these tasks (in advance or automatically at the behest of a user), and the results are somehow made available to the user in an intelligible format, then the user may be relieved of the obligation to conduct multiple time-consuming searches. The consumer may thereafter opt to return to the airlines’ website, or return to the travel website after a designated time, for example, to insert the criteria obtained from a proprietor of the scraping application to obtain the lowest possible fare. None of these activities are currently feasible, however, where the scraping program is simply blocked by the target node.

[0014] As another illustration, a consumer may purchase an asset online at a target website, and an event sometime down the road may trigger the consumer’s entitled to a refund on the asset that the consumer already purchased. In the travel industry, by way of example, prices of assets such as airline tickets may be highly volatile, and hence, possible or likely to change over time. The entitlement to a refund of part of a purchase price may arise, for example, by law, or by a surreptitious provision in an agreement with an eCommerce website. In the conventional scenario, the consumer may not be notified about the discount, and thus may miss out on it altogether. Further, the consumer seeking information about a discount may be relegated to conducting multiple searches of the e-commerce website to establish to what extent, if any, the consumer is entitled to a refund. The average consumer may understandably elect not to pursue these time-consuming tasks, in which case the business owner stands to accrue an additional financial benefit as a result of the consumer’s inability to access information that might otherwise entitle the consumer to a return of some of the funds used to purchase the asset in the first place.

[0015] Countless other examples relating to the utility of legitimate scraping applications in Internet eCommerce and other arenas exist.

[0016] As a result, a need persists in the art for a superior data-retrieval mechanism that overcomes the stated disadvantages.

SUMMARY

[0017] A plurality of agents may be used in a distributed network to perform queries at nodes from which information is desired. A computing system may delegate tasks to perform, such as the execution of queries, to the agents at the nodes. When the tasks are performed, information acquired from performing the tasks may be forwarded to the computing system for storage in a central database.

[0018] A computing system for retrieving data from a node using a plurality of agent computers in a distributed network may include a memory system for storing the code, and a processing system associated with the memory system and configured to run the code, wherein the code when run is configured to deliver a request to each agent computer to retrieve data at the node, receive from the agent computer the data obtained in response to the request; and store the data in a database.

[0019] A computer-program product including a machine-readable medium may include instructions executable by a computing system for gathering data from one or more nodes using a plurality of client computers in a distributed network, the instructions configured to send to each client computer a request to retrieve data at one of the nodes, receive the data from the client computer, the data retrieved by the client computer in response to the request, and store the data in a database.

[0020] A method for retrieving data between a central computing system and an agent computer in a distributed network may include receiving at the computing system an identity of a product, sending to the agent computer a request to perform a search at a node, the search comprising a characteristic of the product, receiving at the computing system data from the agent computer obtained from the search performed by the agent computer in response to the request, and storing the results in a database.

[0021] An article may include a machine-readable medium including machine-executable instructions, the instructions operative to cause a machine in a distributed network of machines to receive from a central computing system a query for data at a node, query the node for the data, and return the data to the central computing system for storage in a database.

[0022] Computers in a distributed network may include a central computing system, and a plurality of client computers, wherein each client computer is configured to send an identity of an item to the central computing system, and execute a search at a node in response to a request received from the computing system, and wherein the central computing system is configured to send to each client computer a request to execute the search at the node, the search pertaining to a characteristic of the item, receive from each client computer data obtained by the client as a result of executing the search in response to the request, and store the data in a database.

[0023] In a distributed network of client computers, each client computer may be configured to receive from a central computing system a query for data at a node, query the node for the data; and send the data to the central computing system for storage in a central database.

[0024] It is understood that other aspects of the invention will become readily apparent to those skilled in the art from the following detailed description, wherein various aspects of the invention are shown and described by way of illustration. As will be realized, the invention is capable of other and different configurations and implementations and its several details are capable of modification in various other respects, all without departing from the scope of this disclosure. Accordingly, the drawings and detailed description are to be regarded as illustrative in nature and not as restrictive.

BRIEF DESCRIPTION OF THE DRAWINGS

[0025] FIG. 1 is a block diagram illustrating an exemplary embodiment of a server system in a distributed network.

[0026] FIG. 2 is a block diagram illustrating an exemplary embodiment of a client in a distributed network.

[0027] FIG. 3 is a flow diagram of an exemplary method for conducting a distributed search.

[0028] FIG. 4 is a flow diagram of a method for conducting a search for a refund.

[0029] FIG. 5 is a flow diagram of a method for rendering an improved results page that, in one embodiment, involves overlaying content onto a web page.

[0030] FIG. 6 is an illustration of a web page from a fictitious travel website overlaid with content.

[0031] FIG. 7 is a flow diagram of a method for initiating and performing tracking as a result of a selection of a user control.

[0032] FIG. 8 is a flow diagram of an exemplary method for conducting a distributed search for a refund.
FIG. 9 is a flow diagram of an exemplary method for creating a tracking rule based.

FIG. 10 is an exemplary graphical user interface (GUI) for configuring a client account.

FIG. 11 is a block diagram of an exemplary embodiment of components of a computing system in accordance with the invention.

DETAILED DESCRIPTION

Agents may be used in a distributed network to perform queries at nodes from which information can be obtained. A computing system may delegate tasks to perform to the agents at the nodes. When the tasks are performed, information acquired from performing the tasks may be forwarded to the computing system for storage in a central database. Because the tasks are performed by the agents, the data retrieval process according to the present disclosure is not thwarted by blocking software at the nodes.

Below is a description of an Electronic Commerce (eCommerce) based application to which the present disclosure may be applied. It should be understood, however, that the present disclosure is not limited to strict monetary-based eCommerce applications. Rather, the principles herein may be equally applied to other areas such as blogs, special interest websites, informational websites, politically affiliated nodes, religious websites, databases, and the like. Further, the principles of the present disclosure are not limited to the HyperText Transfer Protocol (HTTP), but may extend to other protocols and configurations (e.g., file transfer protocol, active server pages, common gateway interface, etc.) whether or not web-based.

eCommerce refers generally to commercial transactions conducted at least partially over the Internet. Users may visit an eCommerce website, for example, using a client device (e.g., a computer, mobile phone, etc.) having a user agent (e.g., web browser, screen reader, mobile phone interface, etc.). While visiting a site, users may search for and purchase “target assets,” or goods and services of interest to a particular user. The ease with which consumers may search for and purchase target assets, such as airlines tickets, hotel reservations, car rentals, cruise tickets, collectibles, computers, books, etc., has contributed to the popularity of eCommerce. Users are interested in having access to timely, comprehensive and targeted information, meaning that users want immediate access to a high percentage of relevant information and a low percentage of irrelevant information.

eCommerce relates to many different economies, of which travel is reported to be the largest and is therefore an appropriate exemplary economy for eCommerce in general. The present disclosure addresses a number of shortcomings of present eCommerce systems and methods in general, and travel eCommerce systems and methods in particular. The present disclosure addresses, for example, the inability to accurately track a target asset over time.

When a user visits a travel website, such as an online travel agency website (e.g., Expedia.com, Orbitz.com, Travelocity.com, CheapTickets.com, etc.), an airline website (e.g., AlaskaAir.com, Continental.com, Southwest.com, etc.), or other types of travel websites (e.g., Kayak.com, Sidestep.com, Priceline.com, etc.), there are a variety of options for searching for information about a current characteristics of a travel asset (e.g., price and availability for an airline ticket, hotel reservation, car rental, cruise ticket, etc.).

However, given that travel asset characteristics are highly volatile, having over a million daily changes to airline data alone, this type of snapshot information may fail to provide a user with adequate information to make an informed decision. Accordingly, users may desire the ability to track a particular target asset over time. A user may, for example, set up a target asset alert at an online travel agency website or airline website and receive periodic updates of “subscription travel information,” where subscription travel information may be defined as the information automatically distributed from a source of travel information. However, subscription travel information may be different than “browser travel information,” where browser travel information may be defined as the information that is delivered in response to a request during a period of interactivity between a user agent and a source of travel information, such as a HTTP response delivered from a web server in response to an HTTP request from a web browser or other user agent. For example, an airline may make its best fares available only on its own website such that a particular fare may be delivered to browsers via browser travel information and not delivered to subscribers via subscription travel information.

Although a user may track a flight by repeatedly visiting a website in order to continually request browser travel information, as noted above, this method for price tracking may lead to user frustration due to the amount of time required to track a target asset. Furthermore, if a user relies on manually searching for target assets and a dramatic fluctuation occurs (e.g., a fare for an airplane tickets drops), the user may be unaware of and unable to take advantage of the fluctuation because the delivery of the information was not timely.

A centralized server system configured to “scrape” information from travel websites may attempt to provide a user with browser travel information by mimicking the functionality of or impersonating a web browser. However, as described above, travel websites may be configured to prevent this type of behavior. For example, a website may be configured to ignore requests from a particular internet protocol (IP) address, a range of IP addresses, and/or a user-agent signature, or take other actions to prevent a centralized server system from providing users with comprehensive and targeted travel information. As described above, the present disclosure describes an apparatus and method for delegating server queries which overcomes this problem.

The present disclosure further overcomes many of the shortcomings of current eCommerce systems and methods as they relate to refund tracking. Because the price of a target asset may change after a user makes a purchase, the user may be entitled to a refund, credit, or other consideration. However, repeatedly checking to see if a price has changed may be time consuming. Furthermore, because airlines may want to prevent high-priced fare purchasers from receiving a refund, determining refund policies and processes might be made intentionally challenging for a user. These and other factors may collectively prevent a user from claiming a refund or credit to which the user is otherwise entitled. The present disclosure describes an improved system and method for tracking a price in order to provide a user with notification of an applicable refund under a variety of conditions.

The present disclosure further overcomes many of the shortcomings of current eCommerce systems and methods as they relate to the inefficiency of using information from a first website at a second website. For example, if a user visits
an online travel agency and finds a desired fare, the fare may include a service fee charged by the online travel agency that the user would not have to pay by purchasing directly from the airline. Having to enter the ticket information (e.g., flight, departure and arrival airports, date, number of passengers, etc.) at the airline website may lead to user frustration. Some users may not be willing to duplicate their efforts, or may be unaware that lower fares are offered at the airline web site, thereby causing the users to pay higher fares for a ticket. Similarly, a user may want to track a fare over time using a third-party website, but having to enter the ticket information of a discovered flight into the third-party site may prevent the user from tracking the fare. The present disclosure addresses these and other problems by providing tag overlay capability as well as dynamic asset tracking.

[0046] Embodiments of the present invention include a method and system for distributed, iterative, and enhanced travel search. Exemplary applications of the present invention include a server system configured to coordinate searches of distributed client applications, a server system configured to track refunds for a purchased asset, and a client overlay tool configured to overlay supplemental content on a certain web pages, such as travel-related web pages.

[0047] Embodiments of the present invention may provide for the following systems and corresponding methods. Specifically, the present invention may provide for managing a network of distributed client executables configured to perform target asset queries. The server system includes a tracking engine that stores what target assets are to be tracked and what users are to be notified when an attribute of a target asset changes. Additionally, the server system has a client coordination engine that communicates with the client executables, including periodically sending tasks to the distributed client executables based on the tracking rules. These tasks include querying a target website (e.g., querying a travel website for the price of a particular ticket) and reporting the retrieved information to the server system. The client coordination engine, as described in greater detail below, may send a task to a first user client despite the fact that this task relates to a second user’s tracking rule such that the first user’s client may be conducting a query on behalf of a second user.

[0048] Additionally, the server system includes a refund engine that allows users to track an asset for the purpose of receiving a refund when a refund is available. By storing characteristics of a purchased asset, the system may periodically query the asset source for updated information about the source and the source’s refund guidelines. By knowing what a user paid for the asset, the current price of the asset, and the guidelines of the source, the refund engine may determine when a rebate is available and notify the user accordingly.

[0049] The server system additionally includes an overlay engine that store overlay rules. These overlay rules determine whether content will be added to a received web page (e.g., whether the page is an overlay page) and what content will be added. A particular example includes overlaying travel information to a results page from an online travel agency website, such as a “buy direct” hyperlink that enables a user to buy a ticket directly from the source without having to re-enter the ticket information and a “track this flight” hyperlink that enables the user to receive periodic updates about changes to the cost of a corresponding flight.

[0050] FIG. 1 is a block diagram illustrating an exemplary embodiment of a server system in accordance with the present disclosure: Shown are server system 200, target websites 112, 113, and clients 101, 102, 103. These devices communicate over a network, such as the Internet 104. For purposes of this disclosure, a client acts as an agent on behalf of server 200 when it performs tasks or runs queries and returns the results of those tasks or queries to server 200.

[0051] The server 200 is a computer or group of computers (further illustrated in FIG. 11) that, depending upon the application, may perform both server and client-related tasks as those terms are used in the art. For clarity, however, the terminology “server” is simply used to distinguish this system from the distributed clients in the network and the target websites.

[0052] Server system 200 supplies the resources (e.g., processors, memory, operating system, etc.) necessary for running a number of engines of executable code to implement the techniques described in this disclosure. Server system 200 includes server engine 201, web engine 202, client coordination engine 203, tracking engine 204, notification engine 205, refund engine 206, and overlay engine 207, as well as communication interface 210. Each of these engines may include hardware, software, power, and networking assets, as described in greater detail below with reference to FIG. 11. Additionally, server system may be implemented on a single server, on a plurality of task specific servers (e.g., a database server and a web server), a plurality of redundant servers, or using other configurations. Server system 200 need not be located in a single location.
an a price variance threshold has been met (e.g., if a user has specified that changes are to be reported for $25 changes and the price has changed by $50, the notification engine 205 may prepare a notification) and whether a preference setting allows for notification (e.g., if a user has specified to receive updates only daily and a notification has already been sent, the notification engine 205 may withhold or defer the notification).

Refund engine 206 stores store refund guidelines for target asset providers as well as target asset characteristics (e.g., purchase price, airports, flight number, and airline). Server engine 201 may create a tracking rule in tracking engine 204 and periodically query target websites 112, 113 (or delegate tasks to clients 101, 102, 103 to query target website 112, 113) to monitor price changes of the target asset. When a price change, refund engine 206 may determine whether a refund is due based on the refund policy or guideline for a particular vendor and the amount of the difference between the purchase price and the current price. If a refund is due, server engine 201 may provide this information to notification engine 205 so that the user may be notified. Alternatively or additionally, the refund engine 206 may simply identify a price change and a reference to the guideline (e.g., a hyperlink to an airline's return policy web page) and allow a user to determine whether a refund is due.

In one embodiment, entitlements to a refund, discount, or other benefit may be stored as one or more tracking rules, which may be stored in the refund engine 206 or in the tracking engine 204. In this embodiment, where the rule authorizes an action, this information can be communicated to a user so that the user can receive the associated benefit. Thus, for example, if the tracking rule is that a refund is authorized by the target asset provider if the price of a target asset changes from $200 to $100, and if the tracked price drops to $100, then this information may be communicated to a user so that he can receive the benefit of the refund (e.g., a $100 credit).

Furthermore, as described in greater detail below, refund engine 206 may claim the refund on behalf of the user automatically or semi-automatically, in accordance with user preferences, global preferences, or other criteria.

FIG. 2 is a block diagram illustrating an exemplary embodiment of a client located in a distributed network in accordance with the present invention. Shown are clients 101, 102, 103 (also referred to as agents), server 200, target websites 112, 113, and the Internet 104. Client 101 includes a client engine 105, web browser 106 with browser overlay tool 107, and server coordination engine 109 and query engine 110 (referred to collectively as client executable 108). Note that, while in this illustration the agents communicate with a target website over Internet 104, a different type of network may be equally suitable depending on the particular application.

Client engine 105 provides processing power for the client device and handles coordination between different applications and engines. Web browser 106 may be a conventional web browser (such as Internet Explorer, Firefox, Netscape, Mozilla, Opera, etc.), a customized web browser for a cell phone, BlackBerry, PDA, or other web interface device. The overlay tool 107 may be a toolbar, such as a toolbar built as a browser helper object for Internet Explorer. Overlay tool 107 extends the functionality of web browser 106 by selectively adding content to a received web page based on a set of overlay rules. In one embodiment, these overlay rules specify that for a given results page (e.g., a results page from an online travel agency), additional content is to be inserted into the output. For example, if a user is searching for flights on Expedia, and the result list that is returned includes ten different flights, the web page may be modified such that the web page that the user sees includes additional content not provided by Expedia. This additional content may include a “buy direct” hyperlink that enables a user to purchase a ticket directly from an airline, and a “track this flight” hyperlink that enables a user to track a flight (e.g., causes a new tracking rule to be created in tracking engine 204). The process of creating overlay rules and manipulating rendered content is described in greater detail with respect to FIG. 5.

Client executable 108 is in one configuration an executable that runs as a process on a client machine (e.g., the client automatically loads the process on start-up without requiring user interaction). In one embodiment, the two components of client executable 108 are server coordination engine 109 and query engine 110. Server coordination engine 109 handles communications of tasks from and results to the client coordination engine 203. The tasks may be, for example, tasks that repeatedly query a travel website for a particular flight until the client coordination engine 203 tells the executable to stop. The task may include a priority, such that if the executable is tasked with several queries, the executable can be directed to conduct a particular task first.

Communications may be initiated by the client executable 108, by the client coordination engine 203, or both. Communications to the client coordination engine 203 may be handled as soon as they are available (e.g., to support a real-time request from a user interacting with web engine 202) or aggregated to limit network traffic or to accommodate communication problems (e.g., updates are aggregated and sent as a group by the server coordination engine 109 and sent to the client coordination engine 203 periodically). Whether a report is sent immediately or aggregated may be dictated based on the initial tasking (e.g., the task may include an immediate response attribute). The actual communication may be by any conventional protocol, including an HTTP request from the server coordination engine 109 to the web engine 202 that can provide the information to the client coordination engine 203.

Query engine 110 executes received tasks based on their priority and in accordance with system and user settings. For example, a system or user setting may specify that there must be five minutes between queries, or query only when client engine 105 is idle, so that a client machine is not burdened by excessive query traffic. In one embodiment, query engine 110 may conduct queries on behalf of multiple clients or users (e.g., not just a user associated with client 101). Additionally, when query engine 110 visits a website, it may provide a user agent signature similar to the web browser 106 on client 101, such that the target websites 112, 113 receive the same user agent signature whether web browser 106 or query engine 110 initiates the request.

Server coordination engine 109 and query engine 110 may communicate via a shared file or files. For example, when query engine 110 receives results from a query, these results may be written to an eXtensible Markup Language (XML) file that the server coordination engine 109 uses to communicate to the server system 200. Similarly, the overlay tool may write to a shared XML file when a user clicks on a “track this flight” hyperlink and the server coordination
engine 109 may use this file to cause a new tracking rule to be created in the tracking engine.

FIG. 3 is a flow diagram of a method for conducting a distributed search in accordance with the present invention. This method includes, in step 302, distributing client executables to a plurality of distributed clients (e.g., clients 101, 102, and 103 in FIGS. 1 and 2). This may be accomplished by allowing users to download software from a website, mailing media to a user, bundling the software with other applications, or by other means. In one embodiment, a user of a client may access the internet to download the executable from a website associated with the server system 200 (FIG. 1).

Next, the user identifies one or more assets to be tracked (step 304). That information is conveyed to the server 200, for example, via the server coordination engine 109 (FIG. 2). Determining a query to be run based on the assets to be tracked (step 306) may involve identifying active tracking rules in the tracking engine and the amount of time since an asset was last checked. In some embodiments, the server 200 simply delivers a request for the client 101 to execute the query. The client executable that is configured to formulate the query, or alternatively to download or receive the query from a user or external source. In other embodiments, as in step 306, the server formulates the query, in this case based on tracking rules, elapsed time since the last search, or similar criteria. In some situations it is advantageous to formulate the query at the server end, to minimize the burden on client resources. In general, the actual formulation of the query may be performed at either the client or server end without departing from the scope of the present disclosure.

Because some websites may not allow automated queries as described above, server 200 may store information about which target websites may prevent server 200 from successfully completing a query of the target website. If the server 200 has been blocked before (decision branch 308) the server system may automatically delegate the query to one or more available client executables (step 310). If the server 200 has not yet been blocked, the server 200 may attempt to query the target website directly (step 312). Querying a website regarding a target asset by a server system may involve a scraping application to request information from the target website for information relating to a target asset. If the query is not successful (decision branch 314) (e.g., the target website fails to respond to the request or responds with information that is different than the information that is delivered to a browser), the system may delegate the query to one or more available client executables 316. Thereupon, after the client executes the query at the target node, the client may send or return the requested data obtained from the query to the server (step 317). The data obtained from the client may be deposited by the server in a central database for future use by the client that ran the query or by other clients in the distributed network. Preparing notification for the user (step 318) may be implemented in accordance with system settings, user settings, tracking rules, or settings for a particular target asset.

In another embodiment, the server system 200 may be set up to immediately delegate the query to one or more available client executables (step 310). This alternative embodiment is illustrated by the dashed line 307 connected to the line extending to step 310, substituting the decision path relevant to a possible server query for a straight delegation instead. The client executable that is the subject of step 316 may be the same client that identified the target asset and target node in the first place. Alternatively, where that client is unavailable, busy, idle or otherwise nonresponsive, the server may delegate the search to another client executable to perform. The latter then will return the results of the search to the server as in step 317. In this embodiment, the tracking or data acquisition process may continue unimpeded even if a requesting client’s resources are unavailable or being used for other applications.

Using the principles described above, a data scraping program is able to access a target website even if the website is running a standard blocking program to block data scraping. Because the actual queries for data are being conducted at agent machines instead of the server 200 itself, the blocking program is unlikely to block access to the target website for the reasons described above. Further, unlike a server running a scraping program that potentially seeks voluminous amounts of data, each agent in one embodiment conducts queries for a relatively small amount of data in comparison. Accordingly, the agents are unlikely to be flagged and blocked by the target website as exceeding a volume threshold at the website. After the data is obtained from the target website by each agent to which tasks are delegated, the resulting data can easily be streamlined and centralized by the server 200 in a database for subsequent use.

In another embodiment, the search by the client may be tailored to have substantially the same characteristics as a search performed in real time by an individual. For example, the search can be made to appear random to the target website in the same way a user may send ostensibly random queries to the website.

FIG. 4 is a flow diagram of a method for conducting a search for a refund in accordance with the present invention. Receiving a set of information at the server relating to a purchased asset from a user (step 402) may involve receiving an airline and a confirmation code, for example, which may be used to determine a variety of asset attributes, including arrival and departure cities, dates, and times, and a purchase price. This information may be stored in refund engine 206 (FIG. 1). Based on the arrival and departure cities, dates, and times, a tracking rule may also be created and stored in the tracking engine 204 (step 404). The tracking rule(s) may delineate criteria for tracking, such as how frequently a follow-up query should be done, at what locations (e.g., websites or databases) the query should be performed, the circumstances under which tracking should continue or terminate, and the like. The tracking rules may also identify whether a user is entitled to some type of benefit.

For example, a prospective purchaser using a client machine to track the price of a target asset may, under the guidelines of the target website or as provided by law, become entitled to a discount. The user’s entitlement to a discount may arise, for example, if the price of the target asset drops below a certain amount. Likewise, a user of the machine who already purchased the target asset may under certain conditions be entitled to a partial or full refund as a result of the price of the asset dropping below a specified threshold. In either case, the criteria for the benefit and an identity of an action authorized in the event of a change in price or other contingency can be stored as a set of tracking rules along with the other types of more typical tracking criteria referenced above.

The tracking then proceeds over time based on the criteria set forth in the tracking rules. Periodically or at des-
For example, as long as the date of a flight for a purchased airplane ticket has not passed, or the user or the system has not deleted or disabled a tracking rule, the system will continue to determine whether a refund is available (steps 410 and 414). This tracking rule may lead the user to periodically checking the price of a flight (step 408), the server delegating a task to one or more client executables to check the price of a flight (depicted in FIG. 3), or other action. If a lower fair threshold is reached (decision branch 410) based on the threshold preferences associated with the tracking rule, the user will be notified (step 412) in accordance with the user’s notification settings. Where the lower fair threshold indicates, based on a rule from a vendor or other criterion, that some action (e.g., issuing a refund) is authorized, then the notification in step 412 may include a notification to the user, a request for the user to confirm that the server 200 can process the refund, or both.

Where it is determined that the purchased asset is no longer active as described above, the server will remove the tracking rule(s) associated with the purchase of that asset from the tracking engine 204 and the tracking process of this example is complete (steps 416 and 418).

In one embodiment, the refund may be handled automatically (e.g., where the user receives a refund if a refund becomes available without user activity), semi-automatically (e.g., where the user is presented with the option of collecting a refund and, if the user elects to collect the refund, the system handles the collection of the refund), or by other means. The billing for this service may be implemented on a flat-fee basis, a percentage of savings basis, a percentage of price of the total asset, or by other criteria depending on the nature of the service.

In another embodiment, the tracking rules may include a rule obtained from the vendor of a product or service. Such a rule may include, for example, a set of criteria for determining whether a particular action is warranted—e.g., whether purchaser of the good or service is entitled to a refund, whether a store credit becomes available, etc. When a change in the price occurs, the server may compare the change in the price (e.g., the new low price or the amount of the change, or both) with the rule. In the case of a refund, if a rule is met, the server may issue a notification to a user identifying that a refund is available (step 412), or handle the refund automatically or semi-automatically as described above.

Rules from the vendor may be obtained from the server by the vendor, from the client, or through other means. The rules may also originate from a vendor of the asset at issue, but may be received by the server 200 as a result of a request to a client machine to execute one or more appropriate queries at the vendor’s web site. Alternatively, the server 200 may execute the query. In addition, the operators of server 200 may provide these rules to the server in advance, based on, for example, provisions of law, website or vendor guidelines, or rules of purchase of various assets from the vendor or other target asset provider. Thereupon, either the server 200, the operators, or another system or third party can monitor the sources of the rules for any applicable changes and the rules in the server can be updated as necessary.

In one embodiment, a user’s entitlement to a benefit like a refund (for an asset already purchased) or a discounted price (for an asset for prospective purchase) is determined by the rules at the server, so that the user is relieved from having to directly engage in the often complicated endeavor of figuring out his or her entitlement, if any, to such benefits. In addition, the computing resources required for making such determinations may be kept in this embodiment at the server 200 to avoid burdening the client machine from having to perform computations relating to this inquiry.

FIG. 5 is a flow diagram of a method for rendering an improved results page that, in one embodiment, involves overlaying content onto travel-related web pages. Receiving a set of overlay rules at a client (step 502) involves first analyzing the results returned by a target website and determining patterns that may be used to overlay content on particular web pages. For example, if a search is conducted on an online travel agency website, a results page may be returned. This web page may have a particular HyperText Markup Language (HTML) pattern, such as a first table having navigation information (e.g., queried flight times and dates) and a second table having a number of different flight results. A rule may be created that identifies each of the matching flights by traversing the document object model, regular expression matching, or any number of other techniques for recognizing data values embedded in patterns in web pages or other electronically-rendered documents. These rules may then be encoded and distributed to client machines having software configured to implement the overlay rules (e.g., overlay tool 107). Thus, an exemplary software application customized for distribution to client machines may include a list of web sites or network locations in which overlay can be performed according to the principles herein.

Receiving a web page response at the client (step 504) involves a user visiting the network location or target website (e.g., the online travel agency website) and conducting a search. Although the flight information may be different, the returned results page may have a format matching an overlay rule. A determination as to whether the web page or rendered document may be overlaid is made (decision branch 506). This determination of whether a page may be overlaid may be made based on some or all of the URL of the requested page, the document object model of the page, a combination thereof, or other criteria. If it is determined that the page may be overlaid based on one of the overlay rules, the content is modified (step 508) and the modified web page is rendered (step 510). For example, if it is determined, based on the domain or the response and the document object model of the HTML page, that one or more flights have been returned, an overlay tool 107 or other process or application may manipulate the rendered web page by adding one or more additional controls to each of the returned flights. If the web page is not an overlay web page, then the server will not modify the content of the web page and the web page may instead be rendered as it was received (step 510). The overlay cycle has thus been performed (514).

FIG. 6 shows an illustration of a website of a fictitious travel agency overlaid with content. The web page is a “results” web page in this example. That is, the web page includes a list of search results displayed in response to a user query. In particular, a user visits the home page of the fictitious agency and executes a query for round-trip flights on particular days (here, Nov. 20, 2007 and Nov. 27, 2007, respectively) on a specified airline (i.e., Acme Air). The website may respond by returning a list of available departure flights and return flights on the identified airline on the identified dates (not shown). Thereupon, the user selects a par-
ticular departure flight and a particular return flight. The web page shown in FIG. 6 displays the results of the user’s selection of these two flights. This exemplary web page also shows departure city (604), arrival city (606), airline (608), travel time (610), cabin class (612), and round trip price (614).

In this example, the overlay tool 107 may provide two or more controls for the first flight (only “track this flight” 615 is shown) and two additional controls 616 and 618 for the second flight. The “buy direct” flight in this example is only shown in the return flight field, because this control may contemplate that the user desires a single purchase for the entire round trip ticket. However, other types of controls or configurations may be equally suitable. As noted above, in the example of FIG. 6, the overlay tool 107 provides two user controls near the data field (620) under the price column (614). Each of the additional controls will be uniquely assigned to the return flight, such that clicking on “track this flight” (616) for the first flight will cause the return flight to be tracked and will not cause the outbound flight to be tracked. In one embodiment, clicking a “buy direct” (618) control from the online travel agency will cause a web page to be rendered that is in a different domain than the domain of the online travel agency (e.g., www.OTA.com is the domain of the online travel agency and www.airline.com is the domain of the airline). These controls are placed near the values with which they are associated (in this example, the results of the query) so that the user can readily identify the controls and select them as necessary. Thus, in the example of FIG. 6, the user controls 616 and 618 are substantially adjacent the price of $138.01 such that a user is more likely to associate the user controls with the corresponding price and the return flight than if the user controls were not placed near the price (but rather, for example, were placed on the top of the page above the list of results and could not reasonably be visually associated with a specific result).

In another embodiment, a user’s selection of or clicking on the user controls placed by the client executable will initiate the tracking process described above with respect to FIG. 4. This embodiment is shown in FIG. 7. At step 702, the client receives a body of executable code as discussed with respect to previous embodiments. This executable code may be configured to communicate with the server. That is, the code may be used to coordinate tasks with the server to accomplish one or more of the objectives set forth in this disclosure. The code in one embodiment may include a software application or suite configured to enable a client in a distributed network to track a different types of data searches executed at a node, such as a website or a remote file server. The code may also be configured to monitor a user’s searches (upon the user’s authorization) and transmit applicable data obtained from searches to the server 200 for processing (e.g., the identification of an asset or a node, or both).

At step 704 the user executes a query for data at a desired node. Meanwhile, the client executable runs in the background and is alerted to user actions which authorize the client executable to perform corresponding actions. Step 704 may be accomplished when a user is searching web pages in a web browser. The code may monitor the searches, and may also prompt the user with a request, in the web browser or separately, to enable tracking features or placement of user controls. Thereupon, the user may click on the request. In other embodiments, the tracking features are automatic and no further user action is required to initiate them.

When the user has obtained a search result, the client executable places one or more user controls strategically positioned near one or more data fields associated with the search results. Illustrative user controls are depicted in FIG. 6 above. If, for example, the data value represents the price of a stock, a control such as “track this stock price” may be placed adjacent the data value for the user to readily perceive. The user control may comprise, for example, a hyperlink, upon which actuation by the user notifies the client executable to track the value in the applicable data field.

In step 708, the user actuates the control by selecting or clicking on the link. At which point, the client executable is prompted to deliver a message notifying the server of the asset (in this case the stock price), along with a request to track the asset (step 710). Thereupon, the normal tracking procedures are commenced by the server, such as those illustrated in FIG. 4. Further, the user may elect to create, accept or modify tracking rules already stored or to be stored on the server 200 in any number of ways depending on the capabilities of the client executable and the server, such as identifying one or more nodes to track the asset, notify threshold values required for the server to notify the client of any change, identify a frequency to track the asset or to notify the client, etc. (step 712). In one embodiment, these activities are automated by the server as a set of default activities, which a user may adjust or delete, as appropriate, through an interface provided by the overlay tool 107 (FIG. 2).

At step 714, during the course of tracking an asset, the client and/or server may collaborate to provide a summary of tracking results or other data on an electronic document such as a web page. The user can then download the appropriate web page and view the results or updates, obtain a refund, or make adjustments to preferences and the like. In other embodiments the client executable is responsible for generating an accessible electronic document without further server intervention. Alternatively, the tracking results or summary may also be provided in a field of the overlay tool 107, an e-mail, or other suitable means.

As stated above, in a preferred embodiment, users may modify the overlay behavior by adjusting their preferences. For example, a user may elect to suppress overlay behavior on a particular page, for a particular site, or otherwise change the way in which the overlay tool operates. This modification may be implemented via the web engine 202, the overlay tool 107, or by other means. Furthermore, the content being provided may be modified based on tracking rules. For example, if a user is currently tracking a flight, and the user conducts a search that returns the flight being tracked, the overlay tool may not provide the “track this flight” control, and may provide another control in its place (e.g., “stop tracking this flight” or “change your tracking preferences”). These preferences may be stored on the client computer or, alternatively may be uploaded to the server for storage and control.

FIG. 8 is a flow diagram of a method for conducting a distributed search for a refund. This method is a combination of the methods described with respect to FIGS. 3 and 4. A user receives application software in the form of executable code. The software may be distributed by the server via a file or files via a download (step 804). Alternatively, the software may reside on a CD, DVD, or it may be obtained from another web site or network location. Using the software, the client enables the server to receive information about an asset that was purchased (step 806). For instance, if
a user purchased a flight on a travel website, the software may enable the client to communicate information regarding this purchase to the server. At this point, based on information transmitted by the client (such as, for example, desired price decrements required for client notification, etc.), the server creates one or more tracking rules for the identified asset (step 808). The server 200 may store the tracking rules in the tracking engine 204 (FIG. 1) as described above.

At decision branch 810, the server inquires whether the purchased asset remains active (e.g., whether a refund is still possible, whether the flight has already occurred, and similar types of criteria). If the asset is no longer active, the server 200 removes the tracking rule from the tracking engine 204, and the exemplary process has completed (814 and 826). If the purchase asset remains active, the server may attempt to conduct the query as in previous embodiments (step 812). If the query is blocked by the target websites (branch 816), then the server 200 may delegate the query to one or more client executables (step 818). After the query is run by either the server or the client, a comparison between the new price (if any) and a user-imposed threshold is made (branch 820). If the price has dropped below the threshold, the entitlement to a refund (if available) is reported to the user (step 824). If not, step 810 through 824 repeat until the purchased asset is no longer active.

As noted above, in one embodiment the server may not perform the query, but may automatically delegate it. This technique is advantageous in situations where automated searches are often blocked by the applicable target nodes as described earlier in this disclosure. In this situation, the agent (client) computers in the distributed network may perform the searches and return the results to the server. In still other configurations, the server may delegate a search requested by one client to another, such that when the requesting client is busy, idle, or otherwise unresponsive, another available client in the distributed network can perform the search.

FIG. 9 is a flow diagram of a method for identifying a target to be tracked. In one embodiment, the target asset parameters are received (step 902) at the server 200, such as a flight number, flight dates and times, and departure and arrival cities. Ranges may be provided, for example, for the flight dates if the user's schedule is flexible. This information may be provided in various ways. One way is through the client executable. For example, a user may authorize the client executable to store this information upon the user selecting a link as described above with reference to FIG. 7.

At step 904, characteristic variance thresholds are received, such as the price variance that occurs before the system notifies the user. For example, the user may select that notifications will be generated only for changes greater than $25. Reporting frequency may be determined at step 906, and a user may select to receive notifications once a threshold is met (e.g., immediately), once per day, twice per week, etc. Once the server has this information, it may create a tracking rule in the tracking engine 204 (step 908), which may then lead to tasks delegated to a plurality of the client executables as previously described. In one embodiment, each new tracking rule is provided with a default set of characteristics (e.g., reporting frequency and price threshold), but the user may override these default settings either globally (e.g., change it so that all tracking rules created in the future will take this new setting) or just for a particular asset (e.g., change the settings for one tracked asset, but new tracked assets will be provided with the system default).

FIG. 10 is an exemplary graphical user interface (GUI) for configuring a client account 1000. This GUI is generated by web engine 202 (FIG. 1) and may be rendered by web browser 106 or other application. Refunds tab 1001 is used for tracking rebates on assets already purchased, while travel planning tab 1002 is used for tracking assets that have not been purchased. The airline dropdown 1003 may be presented to a user first, because the information presented in panel 1004 may be dependant on the airline selected. For example, some airlines may have different information that is required prior to being able to track a flight such that confirmation code 1005 and last name 1006 may be sufficient on one airline, but another airline may require a user to provide departure and arrival date, time, and city (and purchase price for refunds). Hit the submit button 1008 causes a request to be initiated that is handled by web engine 202 (FIG. 1).

In one embodiment, a travel profile may be compiled for each of the users, including frequent flier numbers, travel preferences, previous flights, etc. In this way, the system may continue to provide the user with more relevant information and less irrelevant information without requiring the user to manually update his or her preferences explicitly.

In another embodiment, a user may be provided with customized advertisements based on travel information that a user is tracking. For example, in a notification that a fare has been changed for a particular flight, the user may also be informed of availability and pricing for a nearby hotel.

FIG. 11 shows a block diagram of a computing system used to implement the software functionality in accordance with an embodiment of the present disclosure. The computing system 1112 includes one or more processors 1120a, 1120b, and 1120c. The three computers 1120a, 1120b, and 1120c are networked together through a router 1108 or other network hardware device. While a networked connection is shown here, this configuration is not critical to the disclosure and the computers 1102 may be connected through any conventional means. The networked connection is shown to emphasize that the computers may, but need not necessarily, be in the same location. Each computer 1120a, 1120b, and 1120c includes one or more central processing units (1104a, 1104b, and 1104c, respectively) and memories (1106a, 1106b, and 1106c, respectively). The central processing units 1104 are collectively referenced as a processing system, and may as here include one or more processors. Likewise, the memories 1106 are collectively referenced as a memory system.

In addition, computing system 1112 includes a storage 1110 (e.g., one or more hard drives) for storing data or code obtained from or used by computers 1120a, 1120b, or 1120c or from another source via external network 1114, also coupled to the router. The storage 1110 may, for example, include the client executable for distribution to clients over the external network 1114. A database 1114 is also coupled to router 1108 and may be used by the computing system as a central repository to store data obtained from tasks performed by a plurality of distributed agents in network 1114. In one embodiment, database 1114 is a high capacity, high speed networked array of disk drives.

The processing system 1104 is coupled to the memory system 1106 in the sense that information in one of the memories 1106a-c may store data that can be used by or in conjunction with one or more of the central processing units 1104a-c. The use of the word "coupling" in this disclosure does not require a direct connection between any given
central processing unit and memory. Nor does the use of word “coupling” require that a particular central processing unit must be on the same machine or network as a particular memory.

For purposes of this disclosure, the software and applications run by the processing system 1104 of FIG. 11 and the server 200 of FIG. 1 is also referred to as executable code, server executable code or server code. Similarly, the software or executables run by one or more of the client machines 101, 102, and 103 (FIGS. 1 and 2) is also referred to as executable code, client code, or client executable code. For purposes of this disclosure, the processing system need not be limited to what is referred to as central processing units, but may embody any number of alternative types of processing circuits, whether digital or analog and whether dedicated or general purpose, etc. Likewise, the memory system may comprise any number of topologies such as a random access memory, read only memory, programmable memory, flash memory, hard disk drive, optical drive, etc.

For purposes of this disclosure, the computing system may comprise one or more computers. They may incorporate any of a variety of commonly employed physical and functional server architectures. The computers within the system 1112 need not be in the same physical location and may communicate using one or more wired or wireless networks. Each computer may be dedicated to a single task or function, or alternatively, the computers may split the resources of a plurality of functions. Each computer in the system 1112 may include its own storage, or rely on a central or remote storage repository. In some embodiments storage 1110 is not necessary.

The server executable code may likewise be resident on a single computer 1102, or it may be run on a plurality of computers. Similarly, the server code may be configured to run on one or more CPUs. The memory system contained in computer system 1112 may include a single memory 1106, or it may include a plurality of memories associated with the same machine or with different machines.

The database 1114 may be physically realized as a central repository or the database may be physically distributed over a plurality of locations. The database 1114 may, for example, be configured as a Storage Area Network, or one or more RAID arrays.

The machines described herein may be implemented using software, hardware, or a combination of both. By way of example, the server or agent machines may be implemented with one or more integrated circuits (IC), either alone or in common with other processing functions (e.g., a data processor, etc.). An IC may comprise a general purpose processor, a digital signal processor (DSP), an application specific integrated circuit (ASIC), a field programmable gate array (FPGA) or other programmable logic device, discrete gate or transistor logic, discrete hardware components, electrical components, optical components, mechanical components, or any combination thereof designed to perform the functions described herein, and may execute codes or instructions that reside within the IC, or outside the IC, or both. A general purpose processor may be a microprocessor, but in the alternative, the general purpose processor may be any conventional processor, controller, microcontroller, or state machine. The machines may also be implemented as a combination of computing devices, e.g., a combination of a DSP and a microprocessor, a plurality of microprocessors, or one or more microprocessors in conjunction with a DSP core, or any other such configuration.

The code or instructions may be embodied in one or more machine-readable media to support software applications. Software shall be construed broadly to mean instructions, programs, code, or any other electronic media content whether referred to as software, firmware, middleware, microcode, hardware description language, or otherwise. Machine-readable media may include one or more electronic files, including a set of executable code in whatever format. For example, files comprising a software application downloaded from the Internet constitute a machine-readable media. Machine-readable media may also include storage integrated with a processor, such as might be the case with an ASIC. Machine-readable media may also include storage external to a processor, such as a Random Access Memory (RAM), a flash memory, a Read Only Memory (ROM), a Programmable Read-Only Memory (PROM), an Erasable PROM (EPROM), registers, a hard disk, a removable disk, a CD-ROM, a DVD, or any other suitable storage device. In addition, machine-readable media may include a transmission line or a carrier wave that encodes a data signal. Those skilled in the art will recognize how best to implement the described functionality for the searcher 304. Moreover, in some aspects any suitable computer-program product may comprise a computer-readable medium or machine-readable medium comprising codes relating to one or more of the aspects of the disclosure. In some aspects a computer program product may comprise packaging materials.

The previous description is provided to enable any person skilled in the art to practice the various aspects described herein. Various modifications to these aspects will be readily apparent to those skilled in the art, and the generic principles defined herein may be applied to other aspects. Thus, the claims are not intended to be limited to the aspects shown herein, but is to be accorded the full scope consistent with the language claims. All structural and functional equivalents to the elements of the various aspects described throughout this disclosure that are known or later come to be known to those of ordinary skill in the art are expressly incorporated herein by reference and are intended to be encompassed by the claims. Moreover, nothing disclosed herein is intended to be dedicated to the public regardless of whether such disclosure is explicitly recited in the claims. No claim element is to be construed under the provisions of 35 U.S.C. § 112, sixth paragraph, unless the element is expressly recited using the phrase “means for” or, in the case of a method claim, “step for”.

What is claimed is:

1. A computing system for retrieving data from a node using a plurality of agent computers in a distributed network, comprising:
   a memory system for storing the code; and
   a processing system associated with the memory system and configured to run the code, wherein the code when run is configured to:
   deliver a request to each agent computer to retrieve data at the node;
   receive from the agent computer the data obtained in response to the request; and
   store the data in a database.

2. The computing system of claim 1 wherein the code is further configured to distribute a set of client executable code
to the agent computer for execution at the agent computer, the client code configured to send to the computing system the data retrieved by the agent computer in response to the request.

3. The computing system of claim 1 wherein the retrieving the data comprises executing a query for the data.

4. The computing system of claim 3 wherein the code is further configured to formulate the query.

5. The computing system of claim 1 wherein the code is further configured to receive from the agent computer an identity of an asset, wherein the data to be retrieved comprises a characteristic of the asset.

6. The computing system of claim 1 wherein the code is further configured to:

   attempt to retrieve the data at the node prior to the delivering the request; and

   determine that the node has prevented the computing system from retrieving the data.

7. The computing system of claim 6 wherein the code is further configured to store, based on the determining that the node has prevented the computing system from retrieving the data, information identifying that a search was prevented at the node.

8. The computing system of claim 1 wherein the node comprises a website.

9. The computing system of claim 8 wherein the agent computers are configured to retrieve data from one or more web pages on the website.

10. The computing system of claim 5 wherein the asset comprises a ticket for air travel.

11. The computing system of claim 1 wherein the code is further configured to:

   receive from a first agent computer an identity of an asset;

   deliver the request to a second agent computer to retrieve the data at the node, the data comprising a characteristic of the asset; and

   receive the requested data from the second agent computer.

12. The computing system of claim 11 wherein the code is further configured to send the data to the first agent computer.

13. A computer-program product comprising a machine-readable medium comprising instructions executable by a computing system for gathering data from one or more nodes using a plurality of client computers in a distributed network, the instructions configured to:

   send to each client computer a request to retrieve data at one of the nodes;

   receive the data from the client computer, the data retrieved by the client computer in response to the request; and

   store the data in a database.

14. The computer-program product of claim 13 wherein the instructions are further configured to receive from the client computer an identity of an item, wherein the data comprises a characteristic of the item.

15. The computer-program product of claim 14 wherein the one or more nodes comprise a website.

16. The computer-program product of claim 15 wherein the client computer comprises a personal computer.

17. The computer-program product of claim 16 wherein the distributed network comprises the Internet.

18. The computer-program product of claim 13 wherein the retrieving the data comprises executing a query for the data.

19. A method for retrieving data between a central computing system and an agent computer in a distributed network comprising:

   receiving at the computing system an identity of a product;

   sending to the agent computer a request to perform a search at a node, the search comprising a characteristic of the product;

   receiving at the computing system data from the agent computer obtained from the search performed by the agent computer in response to the request; and

   storing the results in a database.

20. The method of claim 19 further comprising distributing to the agent computer, prior to the sending the request, executable code configured when run on the agent computer to perform the search in response to the request.

21. The method of claim 19 wherein the identity of the product is received from the agent computer.

22. The method of claim 19 wherein the computing system formulates at least a portion of the search.

23. The method of claim 19 wherein the computing system receives the identity of the product from another agent computer.

24. The method of claim 19 wherein the computing system receives an identity of the node from the agent computer.

25. The method of claim 19 wherein the node comprises a website.

26. The method of claim 19 further comprising:

   receiving an identity of the product from an another agent computer; and

   determining that the another agent computer is unable to execute the query at a specified time.

27. An article comprising a machine-readable medium including machine-executable instructions, the instructions operative to cause a machine in a distributed network of machines to:

   receive from a central computing system a query for data at a node;

   query the node for the data; and

   return the data to the central computing system for storage in a database.

28. The article of claim 27 wherein the instructions are further operative to cause the machine to send to the central computing system an identity of an asset, the data comprising a characteristic of the asset.

29. The article of claim 27 wherein the instructions are further operative to cause the machine to send an identity of the node to the central computing system.

30. The article of claim 27 wherein the machine receives the instructions from the central computing system over the distributed network.

31. The article of claim 27 wherein the node comprises a website.

32. The article of claim 27 wherein the data comprises a price of an asset.

33. The article of claim 27 wherein the instructions are further operative to cause the machine to execute the query on behalf of another machine in response to a request received from the central computer.

34. Computers in a distributed network comprising:

   a central computing system; and

   a plurality of client computers, wherein each client computer is configured to:

   send an identity of an item to the central computing system, and

   execute a search at a node in response to a request received from the central computing system;
and wherein the central computing system is configured to send to each client computer a request to execute the search at the node, the search pertaining to a characteristic of the item, receive from each client computer data obtained by the client as a result of executing the search in response to the request, and store the data in a database.

35. The computers of claim 34 wherein the central computing system is further configured to formulate the search.

36. The computers of claim 34 wherein each client computer is further configured to send an identity of the node to the central computing system.

37. The computers of claim 34 wherein the data comprises a price of the item.

38. The computers of claim 34 wherein:

- a first client is configured to send to the computing system the identity of the item; and
- wherein the central computing system is configured to:
  - send to a second client the request to execute the search at the node; and
  - receive the data from the second client.

39. The computers of claim 38 wherein the central computing system is further configured to send the data to the first client.

40. A distributed network of client computers, each configured to:

- receive from a central computing system a query for data at a node;
- query the node for the data; and
- send the data to the central computing system for storage in a central database.

41. The network of claim 40 wherein the data comprises a price of an asset identified by the client.

42. The network of claim 41 wherein a first client computer is configured to execute the query on behalf of a second client computer in response to a request received from the central computing system.

43. The network of claim 40 wherein each of the client computers comprises a software application configured to run on the client computers and received from the central computing system for receiving, executing and sending the query.

* * * * *