INTELLIGENT AGENT PARALLEL SEARCH AND COMPARISON ENGINE

The invention provides an object-oriented system for building and deploying intelligent agent-based search and comparison application quickly and easily for retrieving and comparing information of any type for any industry. The core of the invention system is a suite of modular software engines that are suited to rapid development of semi-custom applications, including: an intelligent agent parallel search and comparison engine, a proxy engine that registers saved queries on host sites, and an agent-based engine that pushes data to online forms, web sites, or databases. A series of tools, all accessed from a common interface, are used to create new applications, alter engine performance, add new information sources to the engine, and make other administrative changes without the necessity of relying on individuals with specialized skills, such as programmers or IS personnel. A scaleable architecture enables the development and deployment of applications capable of performing complicated information retrieval tasks on behalf of a consumer or merchant in the area of network-based information retrieval. Advantageously, the intelligent agent based applications can navigate and understand all possible Internet-based sources: WWW sites, Newsgroups, online libraries, FTP sites and text files and can communicate via all standard protocols including http, via SSL, redirection, cookies and any other security mechanisms.
BACKGROUND OF THE INVENTION

FIELD OF THE INVENTION
The invention relates to software engines for information retrieval in a network environment. More particularly, the invention relates to an object-oriented system for rapid deployment of electronic commerce intelligent agent applications, suitable for any industry or business endeavor.

DESCRIPTION OF THE PRIOR ART
The rapid growth of the Internet has resulted in the ready availability of an unprecedented amount of information to people everywhere. In fact, it may be said that the Internet works too well in this respect. While information of almost any type is instantly available, the ever-changing nature of the Internet, and the sheer volume of information of all types available from hundreds of thousands of individual sites, makes searching for and retrieving useful information a daunting and time-consuming task. Novices may find themselves overwhelmed at the prospect, and even experienced Internet users are frustrated by the amount of time it may take to locate and retrieve information that satisfies their needs.

Various methods of automating the information search and retrieval process have been proposed to simplify and expedite the process of finding information online and subsequently organizing the retrieved information coherently. Parallel searching is the practice of searching several different information sources simultaneously for the same type of information. Of course, the practice relies heavily on the automation capability provided by computer and networking technologies. An example of a system for parallel searching is described by R. Kollin, G. Francis, C. Tiano, System for retrieving information from a plurality of remote databases having at least
two different languages, U.S. Patent No. 4,774,655 (September 27, 1988). The system described by Kollin, et al. provides a search interface that organizes a number of commercial databases into broad subject categories. The user chooses a subject category and formulates a search. The system establishes a connection to the appropriate database or databases and translates the user's search statement into the various query languages of the respective databases. The returned search results are downloaded and the user is free to browse the output at their leisure without incurring additional cost for connect time. The described system simplifies and accelerates the process of acquiring online information from a variety of sources. Kollin's system, however, is merely a search interface, it has no search capability of its own; rather it relies on the search engines of the various databases. The system downloads information that has already been pre-formatted into discrete records by the database vendors, thus it lacks the capability to examine information from a side variety of sources and extract the desired information and construct discrete data items from the extracted information. Furthermore, the user is still required to learn a query language, however simple. Additionally, retrieved information is presented to the user sequentially, rendering comparative analysis difficult.

"Intelligent" software agents have emerged as a means of automating and simplifying the process of searching for and retrieving information in digital format. J. Nieten, Apparatus and method for data transfers through software agents using client-to-server and peer-to-peer transfers, U.S. Patent No. 5,944,783 (issued August 31, 1999) describes an apparatus and method for information transfer among software agents operating simultaneously on a digital network. The described apparatus and method provide server-client, client-server, and client-client interaction among single-purpose software agents operating in a network environment. The agents are able to communicate with each other to accomplish a task that exceeds the capabilities of any single agent. The implementation of applications of the described method and apparatus is apt to be highly complex, requiring
specialized programming skills. It would be advantageous to provide an
intelligent agent search and comparison engine that allowed applications to
be developed and deployed rapidly, and that could be maintained and
administered without special programming skills.

One of the innovations made possible by the Internet's rapid development is
electronic commerce, using the Internet as the primary communications
medium in the sale and purchase of goods and services. As would be
expected, agent technology has found its way into the e-commerce realm. G.
shopping at the point of sale*, Communications of the ACM (March, 1999),
describe the use of a personal digital assistant as a client device to search
a variety of information sources to obtain the best price for a product. P.
Maes, R. Guttman, A. Moukas, *Agents that buy and sell: transforming
commerce as we know it*, Communications of the ACM (March, 1999)
describe agents that negotiate the best price for a product on behalf of a
buyer and seller. The described systems are limited to searching for
information on the World Wide Web. They lack the capability of dealing with
the other information processing protocols common on the Internet, FTP
and Usenet for example. Additionally, implementation of these systems is
apt to require a large commitment of time and effort from individuals having
specialized programming skills. Furthermore, the application is limited to e-
commerce. It would be desirable to provide an intelligent agent search and
comparison engine that could interact with all types of information sources
on the Internet. Furthermore, it would be advantageous to have the capability
to rapidly develop and deploy search and comparison applications for any
purpose that can deal with information processed according to any common
protocol.

R. Doorenbos, O. Etzioni, D. Weld, *Method and apparatus for accessing on-
line stores*, PCT Application No. WO 98/32289 (January 17, 1997) and D.
Christianson, R. Doorenbos, O. Etzioni, C. Kwuk, G. Laukhart, E. Selberg, D.
Ward, *Method and system for network information access*, PCT Application
No. WO 98/12881 (September 20, 1996) disclose methods and apparatus
for online shopping and information retrieval. The disclosed software
agents search network resources, notably the World Wide Web, for the
purposes of online shopping and information retrieval. WO98/12881
employs a complex source description language comprehensible only to
those having specialized skill, and it suffers the previously mentioned
deficiency of being applicable only to information in certain formats. WO
98/32289 is a dedicated shopping application and thus is unsuitable for any
other type of information retrieval. It also suffers the deficiency of being
applicable only to information in certain formats. Both of the described
systems require specialized skill and significant time and effort to
implement.

Comparison engines are known on the Internet. Inktomi™ and mySimon™
are notable examples. Both are comparison-shopping catalogs dedicated
to e-commerce applications. It would be desirable to provide a system
incorporating modular search and comparison engines that allows the
rapid development and deployment of customized intelligent agent-based
applications for any type of information in any industry.

SUMMARY OF THE INVENTION

The invention provides an object-oriented system for building and deploying
intelligent agent-based search and comparison applications quickly and
easily for retrieving and comparing information of any type for any industry. In
one aspect, the invention comprises a suite of modular software engines
that are suited to rapid development of semi-custom applications. The
engines include:

- An intelligent agent parallel search and comparison engine capable
of handling complex data and tasks. The engine is customizable, so
that it may be used for retrieval, storage, and management of any type
of data on any subject or in any industry.
• A proxy engine that registers saved queries on host sites, capturing and compiling search results on a periodic basis, thus allowing host sites to balance agent load.

• A gateway engine, constituting an agent-based engine that pushes data to online forms, web sites, or databases, or for formatting the data to other forms of output such as text files or faxes.

In another aspect, the invention provides a series of tools, all accessed from a common interface, used to create new applications, alter engine performance, add new information sources to the engine, and make other administrative changes without the necessity of relying on individuals with specialized skills, such as programmers or IS personnel.

In yet another aspect, the invention provides a scaleable architecture for developing and deploying applications capable of performing complicated information retrieval tasks on behalf of a consumer or merchant in the area of network-based information retrieval. Advantageously, the intelligent agent based applications can navigate and understand all possible Internet-based sources: WWW sites, Newsgroups, online libraries, FTP sites and text files – and can communicate via all standard protocols including http, via SSL, redirection, cookies and any other security mechanisms. At the top level the architecture includes an Http server for serving up static content, a CGI server for serving up dynamic content, an intelligent agent subsystem, a router/proxy server for controlling all systems and processes, and a database subsystem. End users interact with the system by means of a conventional web browser running on a client machine. The invented architecture provides an aggregation of user services and a set of internal administrative services. All tasks associated with the maintenance and operation of applications developed using the invented architecture are automated with minimal human intervention required.

**BRIEF DESCRIPTION OF THE DRAWINGS**
Figure 1 provides a block diagram of the top-level architecture of an intelligent agent parallel search and comparison engine, according to the invention;

Figure 2 provides a Venn diagram of an aggregation of related public and user services provided by the engine of Figure 1, according to the invention;

Figure 3 provides a Venn diagram of an aggregation of related private and administrative services provided by the engine of Figure 1, according to the invention;

Figure 4 provides a block diagram of intelligent agent subsystem architecture, according to the invention;

Figure 5 provides a flow chart of typical sequences of actions taken by users interacting with the engine of Figure 1, according to the invention;

Figure 6 shows a user interface for specifying search parameters using the engine of Figure 1, according to the invention;

Figure 7 shows an interface for displaying results of the search specified in Figure 6, according to the invention;

Figure 8 shows a page of hyper linked channel and topic listings for accessing an administrative control panel, according to the invention;

Figure 9 shows an administrative control panel, according to the invention;

Figure 10 provides a detailed view of an administration menu in the control panel of Figure 9, according to the invention;
Figure 11 shows an interface for adding or modifying a channel, accessed from the menu of Figure 10, according to the invention;

Figure 12 shows an interface for adding or modifying a topic, accessed from the menu of Figure 10, according to the invention;

Figure 13 shows an interface for adding or modifying a topic cache rule, accessed from the menu of Figure 10, according to the invention;

Figure 14 provides a detailed view of a topic cache management section from the administrative control panel of Figure 9, according to the invention;

Figure 15 provides a detailed view of an agent action control section from the administrative control panel of Figure 9, according to the invention;

Figure 16 provides a detailed view of a session management control section from the administrative control panel of Figure 9, according to the invention;

Figure 17 provides a detailed view of a search profile control section from the administrative control panel of Figure 9, according to the invention;

Figure 18 illustrates a search fields control panel, accessible from the administration menu of Figure 10, according to the invention;

Figure 19 illustrates a save fields control panel, accessible from the administration menu of Figure 10, according to the invention;

Figure 20 illustrates a display fields control panel, accessible from the administration menu of Figure 10, according to the invention;

Figure 21 shows a control panel for administering the WWW sites included in a channel, according to the invention;
Figure 22 shows a control panel for adding a new WWW site to a channel or modifying an existing WWW site, according to the invention;

Figure 23 illustrates an organizational rationale for the World Wide Web, according to the invention;

Figure 24 charts a method for searching a WWW site and extracting information by an intelligent agent, according to the invention;

Figure 25 provides a diagram of a method of describing the pages of a WWW site for the intelligent agent of Figure 24, according to the invention;

Figure 26 provides a diagram for extracting information from the various page elements of a WWW site and assembling a data item from it, according to the invention;

Figure 27 illustrates a control panel for establishing the bounding elements of a page from a WWW site, according to the invention;

Figure 28 illustrates a control panel for establishing bounding elements of a continuation page from a WWW site, according to the invention; and

Figure 29 shows a table of extraction rules for a plurality of data fields, according to the invention;

Figure 30 shows a paging sequence control panel, according to the invention;

Figure 31 shows an interface for inserting a page into a paging sequence, according to the invention;
Figure 32 shows an interface for specifying substitution values in search URL’s, according to the invention;

Figure 33 shows an interface for specifying matching rules for linked search and data fields, according to the invention; and

Figure 34 shows an interface for testing and debugging an application, according to the invention.

DETAILED DESCRIPTION

The terms below, where used in the following Description, have been given the accompanying meaning, rather than the conventionally understood meaning:

Channel – A channel is a broad, top-level subject category for classifying the various information sources available on the Internet. Within the context of the invention, the Internet, and particularly the World-Wide Web, are organized into Channels.

Topic – A second level, more specialized subject category. Also sometimes referred to as a “Search.”

Sequence – The sequence followed by an intelligent agent, instructed by navigation rules, parsing rules and page descriptions, as it navigates the pages of and Internet site.

The invention provides an object-oriented system for rapid development and deployment of search and comparison intelligent agent applications for any
type of data in any industry. Advantageous features of the invented system include:

- Intelligent-agent based - core capabilities of applications based upon ability to quickly perform complicated tasks in the area of network-based information retrieval and processing on behalf of a consumer or merchant; extensive set of components.

- Scaleable architecture - application host systems easily configured for hundreds, thousands or millions of daily "hits".

- Internet enabled - applications understand all possible web-based information sources - internet sites, newsgroups, online-libraries, and can communicate via all standard protocols, http, ftp, Usenet, and so on.

- Automated - every phase of the system is automated (minimal, if any, human or manual interface required); includes automation of the tasks associated with the formation of a new application, database table formation and important search parameters, for example; or automation of existing application configuration functions such as new web site inclusion, parsing rules, etc.; also full automation in all facets of the application in operation - site navigation, agent communication, decision making, information extraction, processing and presentation; also includes a fully automated interface for all application administration tasks.

- Ease of use - all parts of the system have a web-based, intuitive graphic user interface (GUI) available for use in the creation, deployment and administration of an intelligent agent application; internal staff and end-client licensee companies need not maintain extensive technical development team nor incur costly consulting fees.
- Platform-independence - underlying software runs on any platform, seamlessly interfaces to most commercial relational databases through SQL or ODBC.

SYSTEM TOOL KIT
The aggregation of enabling technologies within the invented system may be most conveniently viewed as a system Tool Kit. Due to the object-oriented design of the Tool Kit, there exist several layers in the Tool Kit, ranging from individual objects up to a complete application. Each layer provides the building blocks of the layer above it, in keeping with the hierarchic nature of an object-oriented system. Each of the layers is explained in detail below.

Objects
At the lowest level, the Tool kit can be thought to consist of a fundamental set of building blocks, referred to as objects. There are hundreds of these objects in the Toolkit. An example would be a particular rule; or another, a caching algorithm. Each of these is written in the native language of the underlying system architecture.

Components
At the next level these objects are grouped by intended function into Components. For example, all of the individual rules available for extracting information are collected into the Web Extraction Rule Set Component. Likewise, the SQL query routines are gathered into a SQL Library Component. When a new object is created for inclusion into one of these components, all applications that include this component automatically inherit this additional object and its properties. The Tool Kit also provides a set of facilities for building new components or modifying existing ones, called the Component Builders. These facilities have a web-based interface, come built into the Tool kit for every component, and provide an easy, flexible and programmer-free method to build and manage system components.
The more important components include:

- **Web Extraction Rule Set** - these are series of complicated actions that are available for activation for information extraction from a particular site for a particular search field; an agent loads these rules in when visiting a site and looking for particular information.

- **Web Navigation Rule Set** - offers a set of built-in actions pertaining to how a site should be navigated; an agent launched to visit a site will load the defined actions and use for navigation.

- **SQL Library** - set of routines, that interface to most commercial databases, for performing common tasks such as record addition, deletion and insertion, and various query operations.

- **Data Storage Library** - routines available for storage of information obtained via one or more agents; many different schemes built-in to this component.

- **Web Site Identification Rule Set** - a set of built-in parameters that can be activated in describing a site with information of interest; used by an application to determine if a known site should be included in a particular search.

- **Presentation Library** - series of functions available for displaying information obtained during a user-initiated search.


- **Database Archive Library** - set of routines associated with archiving agent-based searches.
• System Monitor Routines - set of routines associated with checking overall health of web servers, database servers and agents in use.

• Agent Performance Methods - set of routines for configuring agent-based actions in an application.

• Web Site Health Methods - set of routines for monitoring web sites used in an application; can be used to monitor parameters such as speed, usefulness, and availability.

**Modules**

Groups of components are assembled, in conjunction with other capabilities to provide Modules. These modules are then available for the formation of an application that can be targeted to any industry.

The major system modules include:

• Incoming Agent Proxy Server

• Live Agent Coordinator - determines if and when agents are to be invoked as part of an application execution, and if one or more is needed, will launch and then monitor necessary set of agents. Can activate or deactivate agents, research non-responsive site, and so on.

• Web Live Pull Agent - an agent that is able to perform a live visit to a web site, i.e. while a user waits, and perform assigned tasks, usually involving information extraction of some sort and subsequent immediate display or use of the returned information.

• Web Archiving Pull Agent - agent that is able to perform a live visit to a web site, and archive (usually via database) any information retrieved, for later use as part of an overall application.
• Archive-Based Search Agent - agent that can search and display results previously accumulated via a Web Archiving Pull Agent.

• Web Push Agent - agent that can send information to another location, via http or ftp methods.

• Registration Agent - agent that can register a search or other task on behalf of a user of an application.

• Agent Stealth Pack - built into any application; provides set of capabilities to any agents for quiet, non-obtrusive activities on remote sites. Minimizes impact on remote site performance.

• Archiving Agent Scheduler - can use local system facilities for crating schedules for agent-based behind-the-scenes activities that need to take place.

• Agent Balancing System - used to monitor and adjust all agent-based actions on host web server(s); watches each server's load and will adjust various application parameters as necessary to ensure results, performance, and other criteria. High Performance Cache — available for any application; useful for creation of temporary buffer of results; can dramatically speed up some applications.

Applications
Modules are the building blocks of an application. An application constitutes a system for tracking of dispersed, "related", web-based data on any subject and in any format and is intended to serve as the core of an e-commerce business or consumer service. Ideally, an application has the built-in ability to be implemented in dozens of configurations so that it can perfectly match a set of requirements dictated by the specifics of the target industry.
In the design of a new application, at least three different design aspects must be addressed. First, the overall role of the application in the business must be understood and defined. This definition forms the Application Framework. This framework will then be uniformly available for any targeted industry.

Secondly, within the above framework, an industry-specific set of parameters must be defined and maintained. These parameters are used as common elements that must be defined anew as the application is applied to a new industry, in that the values used for one industry probably do not apply in another. Examples of parameters that might need to be identified include a set of web targets for the application search agents, a set of arguments that need to be used to navigate a web site, a set of arguments that need to be used to search a web site, or a set of fields to extract from a web site.

Finally, some of these parameters need to have actual values in order for the resulting application to run and provide useful information. From our examples, we’d need to actually define the sites to use in a search within the target industry, or the actual specifics about each possible search field to be used once at the site, or the navigation rules for a particular site.

For example, a design assessment may ask the following sorts of questions:

Q: The information will be presented to the user when?
A: Now, later, either.
Q: How should the date be retrieved?
A: Archived and local search, live search, combination live with cache.
Q: Frequency of the search available?
A: One-time only, continuous.
Q: Method of continuous search?
A: Pull from vendor, push to vendor.
Q: The system will live where?
A: System servers, client servers, combination.

From this assessment, one can see that there are numerous possible combinations of answers that would greatly affect the design of the application, which defines the modules pulled into the application during run-time.

Application Administrator
The IA Application Administrator is a web-based interface to various controls for all aspects of the applications operation. Examples of available controls (where applicable) are cache management including size, timeout, frequency, etc., session management including size and timeout, live or archiving agent operation controls such as web site timeout, maximum wait time, # of retries, caching on/off switch, etc., and a system monitor.

Application Extender
The Application Extender is a web-based interface to the current set of application parameters and their values as they are defined at any time. The administrator or developer can use this interface to add, modify, or delete the values used for any of these parameters.

Many of the elements and subsystems of the invented system, such as GUI's and intelligent agents, may be developed using conventional programming techniques well-known to those skilled in the arts of computer programming and software engineering. Objected-oriented programming languages having cross-platform capability such as C++ and JAVA are especially well-suited for use in developing the programmed portions of the invention. Many of the components, such as the rule sets, may be scripted.

While the invention provides a scripting language, other commonly known scripting languages would also be suitable. Query routines may be developed using commonly known query languages. The invention is suitable for use with most commercial relational database platforms.
SYSTEM ARCHITECTURE

Turning now to Figure 1, shown is a top level architecture of a system for developing and deploying applications capable of performing complicated information retrieval tasks on behalf of a consumer or merchant in the area of network-based information retrieval. The invention comprises a centralized system connected to a network such as an intranet or the Internet. In the current embodiment of the invention, end users access this system via a network connection, using a web browser running on a client machine 10. They can access applications via the home site or through an affiliate server 15 that has forms or links connecting to the main home site. One or more machines running HTML server processes 12 serve up the static content of the home site. In addition one or more CGI servers 13 serve up the “normal” interactive or dynamic content of the various provided services. Another bank of machines exist to serve the special intelligent agent subsystem. The agent subsystem consists of agent servers 14 designed to launch special, optimized, high performance intelligent agent processes that execute the various tasks associated with the public, user services. Finally, the system includes a database subsystem, consisting of a database server 16 running one or more relational database server processes, connected to a very large-scale databases 17. All of these systems, computers and processes are controlled via a special router/proxy computer 15 that serves as an input/output conduit for all requests, load balances the system, and starts and restarts each process as necessary.

The invention provides a set of public, user services 20 (Figure 2), as well as a very comprehensive set of internal, administrative services 30 (Figure 3). User services are further classified as consumer 21, member 23 and merchant 22 level services. Consumer services are intended for random users who find a website powered by the invented system and try one of the services offered. Member services constitute value-added features beyond the consumer services, when the consumer chooses to register. Merchant services provide features for people or companies that represent possible
information resources upon which the provided services may be based. The invention is described herein with reference to exemplary implementations: the first, a web site for searching mortgage rates, where a consumer may quickly and easily fill out a form specifying parameters of the type of loan they are looking for, and the second a real estate web site, where potential buyers may locate properties of interest. The search and comparison engine sends out one or more agent applications to search a prescribed assortment of information sources so that an assortment of loans meeting the user's criteria may be located and displayed them in ranked format. Thus, the user is able to quickly and easily locate a group of lenders willing to provide the desired loan at an attractive interest rate. Therefore, within the context of the exemplary implementation, merchant services are targeted at lenders who may be included in the application's database of information resources. As shown in Figure 2, there is some overlap between all classes of service, while each group of services have features unique to that area as well.

As previously indicated, the invention provides a set of internal, administrative services, as shown in Figure 3. Functions are included for creating, monitoring and modifying public services for consumers and members. Likewise, a similar set of functions are provided for merchant services. Furthermore, a full suite of functions is included for monitor and control of the overall system.

The invented system provides the following advantageous features:

- intelligent agent based - core capabilities of applications are based upon the ability to quickly perform complicated tasks in the area of network based information retrieval and management on behalf of a consumer or merchant.

- Scaleable architecture - application host systems may be easily configured for millions of "hits" daily.
- Network savvy – applications developed with the invented system can navigate and understand all possible Internet-based information sources - internet sites, newsgroups, online libraries, FTP sites, text files - and communicate via all standard protocols including http, ftp, via SSL, redirection, cookies and any other security mechanisms.

- Automation - every task associated with the maintenance and operation of the system is automated, with minimal human or manual interface required. Includes the tasks associated with the formation of a new channels and searches; creation, modification and deletion of agent database tables; configuration operations such as web site inclusion, parsing rules, etc.; also full automation of all facets of the application in operation - site navigation, inter- and intra-agent communication, decision making, and information extraction, processing and presentation.

- Ease-of-use - all portions of the system have a web-based, intuitive graphic user interface (GUI) available for use in the creation, deployment and administration of user services; internal staff and end-client licensee companies need not maintain extensive technical development team nor incur costly consulting fees.

- Platform-independence - underlying software runs on any platform, seamlessly interfaces to most commercial relational databases through SQL or ODBC connectivity.

Referring now to Figure 4, portions of the system architecture are shown in greater detail. The proxy/router server 15 includes an agent launcher 40, an agent traffic controller 41 and a data portal 42. During execution of a search, the agent launcher 40 launches agents 43 to query a number of sites 44. The agents 43 return retrieved data and pass it to the data portal 42. The
database subsystem includes at least three separate databases:

- A knowledge base data base containing site and channel descriptions, navigation rules and parsing rules.
- Data storage database – containing a long-term archive or retrieved information and short-term caches.
- Users database – including registered members and merchants.

When a registered user (member) logs onto the system, user information is directed to the agent launcher 40 from the users database. The initial knowledge base required by the agents 43 to perform a search is supplied by the agent traffic controller 41 and the agent launcher in turn from the knowledge base database. Retrieved results are routed to the display engine for presentation to the user. Additionally, data may be archived or cached in the data storage database.

MODES OF OPERATION

The system possesses several different modes of operation in response to a user-initiated search:

20 **Instant response mode – information immediately sought**

- Live pull – The user initiates a search, the system launches a live Internet search for data and returns results.
- Archived pull - The user initiates a search, the system launches query of local archive for answers.
- Live pull with cache lookup - The user initiates a search, system searches local cache and then adds results of live Internet search.

**Delayed response mode**

- Live pull: One-time – system searches goes out to network and finds matches to saved search. Continuous – system continuously goes out to network and finds matches to saved search.
Archived pull: One-time - system searches local archive and finds matches to saved search. Continuous – system continuously goes out to local archive and finds matches to saved search.

Web push
- System registers with remote sites; collects results pushed back on behalf of user.

USER EXPERIENCE
Figure 5 provides a flow chart of possible actions taken by a user in navigating a site powered by the invented system. Users first come to a home page 50. In formulating a query, the user first selects a channel 51, one of a series of broad high-level subject categories. Subsequently, a Search, or Topic 52 is selected. The selection of Topic choices is specific to the channel selected. After selecting a topic or search, the user specifies the parameters for a query and executes a search 53. The query may be completely new 54, that is, a live search of network resources, or it may be a search of pre-cached results 56. As described above, the system has the capability of searching archived results, cached results and network resources for answers to the query. After results are returned the user may register the query 55, so that the same query may be rerun at regular intervals. Registering a query is a value-added service, available only to registered users, so the user must register as a member before the query can be registered on the system. Once the user’s registration is confirmed 62, the query is added to the system 61. When a registered user logs in from the home page 50, they are directed to an individualized page. From this page, they may execute a search 63, in a manner similar to that of a random user 51 – 54. Results may be immediately displayed 65, or they may be emailed to the registered user 64. E-mail delivery of results is an additional value-added service available only to registered users. In the case of e-commerce applications, product, vendor and price information may be presented in a mulit-frame kiosk page 57 that includes item 58 and
source 59 and buy 60 frames. Additional services include online help 66 and online news 67.

Figure 6 shows an exemplary user interface from a site powered by the invented system. A user may specify a new query 52 or they may retrieve a registered search profile 61. Figure 7 shows an exemplary results page 57 from the previously described mortgage rate finder application. By entering an e-mail address 64, the user may save a query profile. The source frame 59 displays a lender name and a series of item frames 58 display loan terms.

RELATIONAL DATABASE STRUCTURE

As previously indicated, an integral part of the architecture of the invented system is a relational database structure including a knowledge base database, a data storage database and a users database. Presented herein below is a preferred structure for the relational database.

**Core Tables** (one set of these tables per database)

- Channels - each record defines a topic or Channel that represents a grouping of related areas of information on the Internet, created as a convenience to the consumer.

- Searches - each channel is further broken down into one or more Searches, or specific topics found on the Internet.

- Compares - each record is a rule available for any search for comparing results, e.g. CASE-INSENS, BOUNDED-BY, etc.

- Defaults - each record is a rule available for any search for filling in a default value for a field, e.g. DATE, TIME, DATETIME, SEARCH.
Filters - each record is a filter with built-in conversion rules to be used with any value extracted as part of any search, e.g. LC, STR, PRICE, REAL, NUM, PHONE.

Each of the above-named tables is described below in greater detail.

**Table:** Channels

Fields:

- **keyname** - unique, 3-character identifier, e.g. SNIP, ENT, RES, etc.
- **name** - channel label
- **searches** - list of Search table record keynames, separated by '||', e.g. RES||AUC||RAT
- **icon** - name of image to use on application page to designate this channel
- **c1** - unused
- **c2** - unused
- **c3** - unused

Description: Information available on the Internet is grouped into topics which referred to as “Searches”, and related Searches are grouped into a Channel. Each record in this table represents a channel. The “keyname” is used for internal operation; the user never sees this designation. The “name” field is the label displayed on a site and seen and referenced by the user. The “Searches” field is a list of all the Searches belonging to this Channel. Searches may belong to more than one Channel. The icon is the image that also identifies the Channel and may be displayed in various places on a site.

**Table:** Searches

Fields:
- keypoints - unique, 3-character identifier, e.g. AUC, FND, PPL, etc.
- name - label for channel
- icon - name of image to use on page for this search
- c1 - unused
- c2 - unused
- c3 - unused

Description: Information available on the web is grouped into Topics, also referred to as Searches. Each record in this table represents a Search. The "keyname" is used for internal operation; the user never sees this. The name field is the label displayed on a site and seen and referenced by the user. The "icon" is the image that also identifies the Search and may be displayed in various places on a site.

Table: Compares
Fields:
- keyname - unique name for the comparison rule
- txt - rule description, for internal reference

Description: During information retrieval and processing, it is often necessary to do some analysis of the data by way of comparison against something else. Often a series of operations will need to be performed as part of this comparison. Each of these records is a built-in rule available for use during these operations. For example, the record CASE-INSENS indicates that a string-string comparison should be case-insensitive.

Table: Defaults
Fields
- keyname - unique name for the default rule
- txt - description of rule, for internal reference
Description: During information retrieval certain items are to be displayed or saved for further processing. Some of these items come from external sites, and some are internally generated by the system. Any of these items could be set up to use a default value rather than something empirically derived.

The records in this table are the types of default rules available. For example, the record DATE implies that the current date should be inserted as the value for that item.

**Table:** Filters

<table>
<thead>
<tr>
<th>Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>• keyname - unique name for the filter rule</td>
</tr>
<tr>
<td>• txt - description of rule, for internal reference</td>
</tr>
</tbody>
</table>

Description: During information retrieval certain items are to be displayed or saved for further processing. Some of these items come from external sites, and for a variety of reasons, before they are used they may need to be "cleaned up" - extraneous or special characters removed, and so on. These items might also need to be validated that they indeed contain something expected and of use. To aid in these determinations, a set of built-in filters is provided by the records of this table. Each of these is a rule that, if indicated, is to be applied to an item value. An example is the PHONE filter - which can imply removal of non-digits, as well as a confirmation that the value under scrutiny is indeed a valid phone number of some sort.

Each "Search" or topic within a Channel is also described by a set of tables:

**Search Tables** (one set of these tables per Search)

- Sites
- Substs
- SiteSelection
- Display
• Descriptors
• Elements
• Sessions
• MatchRules

5
•Urls
•Types
•SessionItems
•SiteStatus
•Seqs

10
•Fields
•Cache
•Pages
•SearchFields
•Searches

15

The foregoing description of the system database structure is not intended to be limiting. Tables may be added or subtracted according to the application. Furthermore, other database configurations consistent with the spirit and scope of the invention will be apparent to those skilled in the art of database design.

DEVELOPMENT AND SYSTEM ADMINISTRATION CAPABILITIES
The invented system offers a complex set of capabilities available for the formation of powerful consumer web-based search services. These capabilities are automatically available to every Topic or “Search” in every Channel generated within the system. One of the most important tasks involved in the development and development of application from the system of the invention is the creation of new Channels and Topics, also known herein as “Searches.” The various steps required for the creation of
new channels and topics is described below in overview. Each of these steps will be described in detail in subsequent pages. Several of these steps could be performed in any order, so the order presented below is merely exemplary. Other sequences will be apparent to one skilled in the art.

When adding a new topic, if necessary, create a new Channel, then:

- Create the new Topic - this will create the database tables associated with the new topic, modify some code, and create some new libraries for the system.

- go back to the Channel(s) and associate the new Topic with one or more Channels.

- Configure the initial set of Topic parameters (these may be modified later).

- Specify data fields to extract; as each field is specified the cache table is extended to include the additional field.

- Specify the cache rules for items saved via this topic - parameters such as keyname fields, date field, and default fields are specified here.

- Design and register how the data retrieved is to be presented to the user via a live or archived search function.

- Topic won't function unless at least one site exits in the system; add one, then program agent rules for identification, navigation and interaction.
- Modify live engine if necessary - types setup, search parameters and cache extraction rules entry.

- Modify the Topic_Site module, if necessary.

5

The developer or administrator gains access to private and administrative functions through a CGI-controlled access panel. After successful login, a page 80 with a listing of all available channels 81 and their accompanying topics 82 is displayed, as shown in Figure 8. Each topic listing is hyperlink to a CGI program that calls a control panel 90, shown in detail in Figure 9.

10

The control panel 90 represents administrative functions available for every topic on the system. The control panel 90 includes areas for cache management 91, session management 92, agent action configuration 93, query profile administration 94, user query screen configuration 95, banner ad management 96 and an administration menu 97. The administrative control panel, through its several functional areas, constitutes a toolkit for administering existing topics. Upon addition of a new topic to the system, a parallel set of functions are automatically generated and presented to the developer or administrator through a similar control panel, described in detail further below.

15

Additional developer and administrative functions are provided in the administration menu 97, shown in greater detail in Figure 10. In the 'System Components' section of the administration menu, are found links for adding or modifying a Channel 100, and adding or modifying a Topic 102. When the 'Channels' link is selected, a control panel is displayed that allows the addition of a new Channel, or modification of an existing channel; shown in Figure 11. To modify an existing channel, the administrator simply selects the channel name from a pulldown menu 110 of exiting channels. Following selection of an existing Channel, the channel key 112, the channel name 111, and the selected topics 113 included in the channel may be modified. To add a new Channel, the channel key 112 and the channel name 111 are
entered in the appropriate entry fields of the control panel. Subsequently, the new Channel is populated with Topics by adding them from the selection of available Topics 113. As will be explained further below, the administrator may also create a new Topic to add to a Channel. Following Topic selection, clicking the 'Add Channel' button 110 adds the newly created or newly modified Channel to the system.

The most complex aspect of developing and administering applications powered by the invented system is the design and administration of Topics, also known as 'Searches.' The Topics or 'Searches' have a broad range of associated capabilities and attributes. These capabilities and attributes are replicated identically across every Topic, but specifics differ from Topic to Topic. The following description presents in detail the steps involved in creating a fully functional Topic within a Channel. Most of the functionality is built in and inherited by the Topic at each step, but unique aspects must also be established by the topic designer, requiring a thorough understanding of the subject matter represented by the Topic.

To create or modify a Topic, the administrator selects the 'Searches' hyperlink 102 from the administration menu 97. As Figure 12 shows, a control panel 120 for adding and modifying Topics appears. As with the menu of existing Channels, the administrator may choose from a menu of existing Topics 123 to modify a topic. The Topic key 122 and the Topic name 121 may be modified. A 'Delete' button 124 allows for the deletion of a Topic no longer needed. During creation of a new Topic, the new Topic name and the new Topic key are entered into the appropriate entry boxes, and the Topic is added to the system by clicking the 'Add' button.

At the time a new Topic is created, a cache is created for the Topic as well. Certain Data items retrieved from the network during user initiated searches are stored in the Topic cache. Attributes of the cache, and therefore of the cached items are specified by a cache rule for the Topic. The administrator may create a new cache rule or modify an existing one by
selecting the 'Write to cache' link 101 from the administrative menu 97. Following selection of this link, the administrator is presented with a Cache Rule control panel 130, with which the administrator may add or update a Cache Rule. Each cached item is given a unique identifier or key name determined by concatenating the values retrieved for selected data fields in the cached item, specific to the Topic. Creation and modification of fields is described further below. A series of checkboxes 131 is presented, with one checkbox corresponding to each of the Topic fields. Selecting a checkbox includes the value of the corresponding fields in the key name for the cached item. The current date may be inserted into the item by selecting a data field for inclusion of the date. A group of checkboxes 132, each corresponding to a data field, is provided for date inclusion. The field selected will have the date included in the field. Ordinarily, fields are populated with data extracted from various network information sources during a user-initiated search. However, when a field is created, the field may also have a default value specified. A third group of checkboxes 133, each corresponding to a field, allows the administrator to select a field or fields for which the default value is filled-in in advance. Even though a default value may be specified for a field, the default value is not entered into the field unless the field is checked in the Cache Rule Control Panel. A pulldown menu of Topic keys 134 and a Topic ‘Goto’ button 135 allow the selection of a particular Topic to facilitate navigating to the Topic for which the creation or modification of the Cache Rule is desired. After the Cache Rule is specified to the satisfaction of the administrator, clicking an 'Add Rule' button 136 adds the rule to the system. A second menu of Topic key names 137 and a 'Delete' button deletes a selected Cache rule from the system.

As previously indicated, the Administrative Control Panel 90 includes a 'Cache Control' section 91. Figure 14 provides a detailed view of the 'Cache Control' section. The current cache size 140 indicates the number of items currently cached. An 'Empty Cache' link 140 clears the cache of all cached items. A cache management process runs in the background to check the
cache for items that have exceeded the specified age. Controls 142 specify how often the cache is to be checked and the maximum permissible age of cached items. In the example shown, the cache is checked every six hours and the maximum permissible age for any item thirty-two hours. Each topic has an automated process that allows the administrator to pre-topic or pre-archive sites 143 within the system, permitting faster access and less load on remote hosts. A 'Save' button 144 saves the Cache Control Settings.

During execution of a user-initiated query, each Topic makes extensive use of the system intelligent agents. The 'Agent Action' section 93 contains parameters that control agent behavior as the agents interact with remote sites. The maximum permissible time 150 a user should wait for the launch of an agent is specified. The maximum number of times 151 an agent should try a site that is busy or other wise unresponsive is specified. The next control 153 specifies the maximum amount of time to wait for a response from a remote system. 'Display Presort' 152 may be set to 'on' or 'off.' If set to 'on,' the system accumulates all results from all sites before displaying. if 'Display Presort' is set to 'on,' the maximum number of items per site per page and the maximum number of items per page 154 may be specified. Subsequently, the settings are saved.

Every user-initiated search is recorded as a session. A session consists of the query parameters used and the results generated. The session management section 92 provides a mechanism to control sessions as they are generated by the many users of each Topic. The current number of sessions for the Topic 160 is displayed, and all current sessions may be removed 161. The system process may be instructed to check sessions at specific time intervals 162, in this case every hour, and a maximum age for each session may be specified, for example, as shown, ten minutes. A 'Save' button 164 saves settings.

Public services include a user registration entry or 'profile.' The 'Search Profiles' section provides a series of controls for managing these Search
Profiles. The number of profiles for a Topic is displayed 170 and all current profiles may be removed 171. Each profile has an assigned ID, and, generally, each profile includes an action associated with generation of that profile. By entering the profile ID an clicking the 'Go' button, the administrator may execute a profile for testing purposes.

Many user services involve performing a search of network information resources. For any given Topic, the user needs to tell the system the specifics of what they are looking for. Therefore, search fields must be provided that allow the user to adequately describe what information they seek, in order to maximize the possibility that they will find what they are looking for. Clicking on the 'Parameters that are needed to search each site' link grants access to a control panel that allows for the complete definition of each required search field. The search fields are used in a variety of services, including search forms, data integrity checks, and data matching. Figure 18 provides a detailed view of the Search fields definition control panel. Using this Control Panel, the administrator may create new search fields or modify existing search fields. A key name 131 is associated with each search field. Each field has an associated description 180, which is the field label visible to the user. Each field also has an associated internal variable name 181 that identifies the field to the system. The field type is specified 187. In the example the field type is "One_only" meaning the field can be set via user entry. Field type 'One_or_more" indicates a field providing a multiple choice selection. Additional field types are "default" and "unused." "Default" causes the value entered by the user to be used as the default value in a linked data field. For field type "One_or_more" the values and labels for the choices 182 are listed separated by '||.' for field type "One_only," the field size and maximum string length 183 are specified. A pulldown menu 185 and a 'Goto' button allows the administrator to select a current search field to modify. A 'Delete' button allows a current field to be deleted from the system. An 'Add' button 186 saves changes and adds newly created fields to the system. A pulldown menu of data fields 188
allows the search fields to be linked to data fields. Field type 'Unused' functions as a place holder.

Some of the public user services involve performing a search of network information sources, often involving the extraction and overall management of information found on the network, particularly the Internet, involving the creation and administration of data fields. To create these data fields, the appropriate control panel is accessed by clicking the 'Data fields you expect to find' link 104 from the Administration Control Panel. Figure 19 shows a control panel that allows for the complete definition of data fields for a Topic. The example shows a field that allows for a user type-in value, in this case, a real number. As with the Search Fields Control Panel of Figure 18, the Data Fields Control Panel allows the specification of a field key name, a label, a field type, a listing of existing data fields for the topic, a 'Goto' button, an 'Add' button, and a 'Delete' button and menu.

As with individual Topics, a cache rule may be specified for a Channel. To specify a Cache Rule for a Channel, the administrator clicks the 'how the Data is to be archived or Cached' link 108 from the administration menu. The control panel for specifying a Cache Rule for a Channel is almost identical to the Cache Rule Control Panel of Figure 13.

Some of the user services provide for a tabular display of results found via a user-initiated search. The invented system possesses a dynamic display table builder facility that allows a Channel designer to control the appearance and behavior of the results table displayed to the user. The table builder facility is accessed by selecting the 'how the Data fields are to be Displayed' hyperlink 107 from the administration menu. Figure 20 shows a Control Panel for adding and updating data display elements. A separate field is displayed in each column of the table. Each column has a key name 131 that corresponds to the field that occupies the column. Each column has column header label 180 corresponding to the field label. Each entry in a column can be a hyperlink 188 to another location. If the field is to be
linked out, the linking field is selected here. The column width 189 is specified in pixels. The field type is specified from a pulldown menu of field types 187. In this control panel, permissible field type choices are ‘regular’ and ‘image.’ As with other field Control Panels, there are ‘Delete’ and ‘Add’ buttons 184, 186 and a ‘Goto’ menu and button 184.

WEBSITE SPECIFICATIONS AND UPKEEP
A Channel search involves the agent identifying, navigating and searching a series of pre-identified, applicable web sites. These web sites must be identified and categorized for each Channel. The administrative control panel for each Channel provides a special section for the addition, modification or deletion of useful websites into the portfolio that should be available for the agent. This section is accessed by selecting the ‘which Web Sites to search’ link 106.

Shown in Figure 21, from this control panel all of the web sites available to an agent are entered 210, modified 211 or deleted 212. Note that this is the master list of all possible web sites to use - in almost all cases when the agent is launched as part of a user service, it will make some initial analysis and include or exclude some of these sites depending upon a host of sophisticated algorithms designed to optimize all aspects of the overall system operation and the user’s results. This is the panel used to add or modify an existing web site for this Channel. This example is the modification display for the Home Web (RHC) web site. When a Web site is entered 210 or modified 211, the control panel shown Figure 22 is presented. Each website is assigned a site code 221. A menu of existing sites 222 allows the administrator to select another site to modify without returning to the previous control panel. The administrator enters the site URL 220, and the site name 223, and specifies the site type 224. The site status 225 allows the administrator to set a site to ‘active’ or ‘inactive,’ in which case the site would not be searched. The administrator may specify a non-responsiveness threshold value 226 as a quality control measure. The value corresponds to the maximum number of times a site may fail to
respond to a query before a warning is sent to the administrator that the site is unresponsive. Typically, unresponsive sites are deleted from the channel. The control panel also has 'Add' 227, 'Clear' 229 and 'Return' 228 buttons.

AGENT-WEB INTERACTION RULES

The agent is designed to seek out web sites pertaining to a Channel, navigate to pages containing information of interest, and extract this information to be sent back to the system for further processing, display, storage, etc. For each step in this process, the system provides one or more control panels to help define or direct how the agent should behave. The 'how to use the search parameters to find pages on the site, navigate through them, and filter the desired data fields' link 107 is selected from the administrative menu.

Within the context of the invented system, the World-Wide Web is viewed as a collection of information Channels 100. All information sources, or websites 230, i.e. internet sites, FTP, usenet newsgroups, and so on, fall into one or more of these Channels, as shown in Figure 23. Each web site is broken up into one or more search SEQUENCES 231. Each SEQUENCE is defined by a SEQUENCE DESCRIPTION. The SEQUENCE DESCRIPTION consists of a series of PAGE's 232 and the traversal rules between them, and a mapping of the user input search parameters to the PAGE traversal rules. Each PAGE within a SEQUENCE is described with a PAGE DESCRIPTION. The PAGE DESCRIPTION is a collection of PAGE ELEMENTS and their interrelationships, and a mapping of these ELEMENTS to a set of DATA ITEMS. The possible PAGE ELEMENTS are Main Page, Frame Page, Subpag, Continuation Page and Transition Page. DATA ITEM's are comprised of predefined data fields that are extracted from various PAGE ELEMENTS.

WORLD-WIDE WEB: CHANNELS

-35-
In the example agent search shown in Figure 24, there are currently four websites available to the agent - A, B, C and D. Based upon certain knowledge-based rules, the agent determines that only sites A, B and D need to be visited to fulfill the current objective. Site A has been defined to contain two possible SEQUENCE's 240, 241 specified for a search. Site B also has two 242, 243, while Sites C and D each only have one defined SEQUENCE 244, 245.

The first SEQUENCE of Site A is defined as a traversal through 3 PAGE's: P1, P2 and P3, with the latter two being involved in data extraction by the agent. The second SEQUENCE is defined as a traversal through 2 PAGE's P1 and P2, with only the latter PAGE being involved in data extraction.

The first SEQUENCE of Site B consists of 4 PAGE's, the latter two being included in data extraction. The second SEQUENCE has 2 PAGE's with the last one being involved in data extraction.
Both Sites C and D have simple one page SEQUENCE's in which both PAGE's are to be involved in data extraction.

In this example, the agent has further determined that only the first SEQUENCE of Site A should be executed for its current mission. The second SEQUENCE is ignored this time, in future visits this site may not be ignored.

Likewise, the agent will visit the second SEQUENCE of Site B only, and the first (and only) SEQUENCE of Site D.

The interaction between the agent and the PAGEs, SEQUENCEs, and content of each site defined within a Channel is described in greater detail below.

The process of programming an agent for a site involves selecting the site from a pulldown menu (not shown) of all sites registered with the current Channel, and defining pages, sequences, data field match and extraction rules.

A task associated with a Topic is a search for data by the agent of one or more web sites in response to a user query. The agent can perform these tasks because the entire Internet has been analyzed and broken into a collection of conceptual elements. All Internet sites can be considered to fall within one or more topical Channels. Each of these web sites is described within the system framework by a WEBSITE DESCRIPTION. The WEBSITE DESCRIPTION consists of one or more SEQUENCEs. A SEQUENCE is defined to be a series of PAGES, with an implied traversal from one PAGE to another. Each PAGE can conceptually be thought of as a consisting of a series of nested building blocks known as the page ELEMENTS. Each of these ELEMENTS has a set of properties associated with them. The specification of these ELEMENTS and their interrelationship for a given
PAGE is the PAGE DESCRIPTION (see Figure 25). The agent understands how to read and understand a website SEQUENCE. It can also read and interpret the PAGE DESCRIPTION's that comprise the SEQUENCE as part of its built-in expert system on web navigation and information extraction. The agent is sent to a site knowing it is to execute a certain pre-defined SEQUENCE. It knows it must navigate to and traverse each PAGE in the SEQUENCE. For each PAGE, it simply loads in the PAGE DESCRIPTION, interprets it, and executes it.

This section describes how the PAGE DESCRIPTION is formed for any PAGE. Subsequent sections describe the SEQUENCE and other areas that make up the WEBSITE DESCRIPTION. The building blocks available for forming the PAGE DESCRIPTION are the Main Page 250, Frame Page 251, Subpage 252, Continuation Page 254 and Traversal Page 253 elements. All PAGE DESCRIPTION's start with the Main Page 250 element. All other elements of a page fall within the Main Page. The next elements that may exist are one or more Frame Pages 251 in series within the Main Page. Within each Frame Page may exist one or more Subpages 252 in series. If there are no Frame Pages present, the Main Page may still consist of a series of Subpages. The Main Page element should not be construed as having to physically exist as a single web page. It may actually span several physical web pages on the site. For example, each physical web page may also have one or more Continuation Page 254 elements. A Continuation Page is another physical web page that replicates or continues the Frame Page 251 and Subpage 252 sequencing described for the first physical web page encountered. Each Continuation Page 254 may link to another Continuation Page indefinitely. The Transition Page 253 is also possible from within any other element.

The agent's objective in visiting a website is to find some data in response to a query of some sort. This data can be thought to be one or more DATA ITEMS. Each DATA ITEM is comprised of a series of predefined data fields. Part of the PAGE DESCRIPTION is a mapping of the data fields to the PAGE
ELEMENTs. In executing the PAGE DESCRIPTION, the agent attempts to step through the page ELEMENTS, collect data fields as specified, assemble these into matching DATA ITEMS, and return them to the system.

A diagram of the locations and assembly of possible data fields is included in Figure 26. All ELEMENT's, the Main Page 250, Frame Page 251, Subpage 252, Continuation Page 254 and Transition Page 253, can have one or more data fields 262 associated with them. Main Page fields are included with every DATA ITEM assembled from within the entire PAGE. Frame Page fields are included with the DATA ITEM's assembled within that Frame only. Subpage fields are only included with a DATA ITEM 260, 261 assembled from that Subpage. At most, one DATA ITEM is assembled from a Subpage. Transition Page Fields are extracted from the transition web page, which is formed via the special TR data field (explained later). All of these rules are replicated for each Continuation Page, which is assumed to mimic it's preceding page in structure.

A key concept in programming the PAGE DESCRIPTION is the idea of using the HTML source of the current web page as a reference for setting up certain boundaries and rules for some of the ELEMENTS. As the agent is loading in web pages from the site, it has a built-in HTML parser that interprets and parses the source HTML code according to these boundaries and rules.

For example, the PAGE DESCRIPTION may include a Frame Page ELEMENT. This component is a conceptual sub-partition of the web page. As such, its bounds need to be defined; the agent needs to know where the Frame Page begins and ends, as the actions it takes while within it differ from those it takes when it is outside of it. The bounds for some of the other PAGE DESCRIPTION ELEMENTS are also needed, and are an important part of the agent setup. To easily enable the establishment of the bounds for an ELEMENT in a text file (which, after all, that is what the agent is working with), a key concept is the establishment of a bounding element. In
programming. a bounding element is a set of parameters which, when interpreted and applied to the text file under consideration, establishes a beginning location and ending location for an item. A bounding element consists of up to six components - the 'value,' 'start,' 'begin,' 'begin offset,' 'end' and 'end offset.' For example:

Position in file for an ELEMENT = bounding element
bounding element = text from START to END
START = position(start) + position(begin) + begin offset
END = position(end) + end offset
start, begin, end = TEXT REF . begin, end = POSITION.

The 'start,' 'begin' and 'end' components are TEXT_REFs. A TEXT_REF is a rule that points a parser at a certain location in a text file. The simplest form of TEXT_REF constitutes one or more characters. If there are no embedded commands (the complete set of TEXT_REF and POSITION rules are included below in the section PARSING RULES) the parser simply looks for the characters in the text file and returns the position at which it found them.

The 'begin offset' and 'end offset' components are numerical values (but may also have embedded commands), used to increment or decrement the resulting positions found via the 'begin' and 'end' elements, respectively.

Bounding elements are nested within ELEMENTS of the page. They are referenced from the starting location of the current ELEMENT, rather than from the start of the actual web page. For example, a bounding element defined within a Subpage, is referenced from the beginning of the Subpage. Or, the bounding element defining the start and end of a Subpage itself is referenced first from any bounding Frame Page ELEMENT, and second, third, etc. to any prior Subpages.

Bounding elements are employed in several places in the PAGE DESCRIPTION specification. Their use as part of the PAGE ELEMENT's has
just been described. They are also used to establish the bounds of a piece of text to extract in other places. For example, bounding elements are used as part of a data field extraction rule - the data field boundaries are established via a bounding element, and the initial value for the data field set by extracting the text within these bounds, or as the specification for establishing the continuation indicator, described in greater detail below. For these latter instantiations, the bounding element ‘value’ field comes into play (it is ignored for the first case described). If this field is set, the bounding element is interpreted as returning this value - the other parameters are ignored.

Yet another use of the bounding element is as part of the ‘search URL’ construct, another key component of the PAGE DESCRIPTION, and presented next. ‘Search URL’ is a construct that results in an Internet address, or URL, of the web page on a web site being created, which will presumably be loaded in at some point by the agent. This construct has an http method indicator, GET or POST, and six bounding elements. The first three - the ‘tag prefix,’ ‘tag body’ and ‘tag suffix’, are used to form the address of the URL if it is a POST, and the entire URL if it is a GET. The second three - the ‘arg prefix’, ‘arg body’ and ‘arg suffix,’ are used to form the argument list if POST, and not used if GET. Each of the ‘tag’ and ‘arg’ element fields are extracted from the source page and concatenated together to form the URL for an http call - all shown Figure 28. All of the “normal” bounding element rules apply.

Method GET:
URL = Tag Prefix + Tag Body + Tag Suffix ; Tag Prefix, Tag Body, Tag Suffix = bounding element

Method POST
URL = Tag Prefix + Tag Body + Tag Suffix +? + Arg Prefix + Arg Body + Arg Suffix
Tag Prefix, Tag Body, Tag Suffix, Arg Prefix, Arg Body, Arg Suffix = bounding element

Referring now to Figures 27 and 28, two control panels are shown that enable the developer or administrator to define a page description. Figure 27 shows how to define the existence of Framepages and Subpages. A 'yes' or 'no' button 270 allows the developer to specify that a Framepage is present. Controls 271 allow the Framepage bounding elements to be defined. Another yes' or 'no' button 273 allows the developer to specify that a Subpage is present. Controls 272 allow the Subpage bounding elements to be defined. A listing 274 of the Subpage data fields is also displayed. The control panel of Figure 28, allows continuation pages to be defined. A 'yes' or 'no' button 284 indicates whether a contination page is possible. A text box 280, allows the specification of the continuation indicator. the http access method is indicated 282. The bounding elements 281 that created the URL for the http call are specified. Subsequently, the continuation information is saved 283.

On every PAGE, it is possible to extract one or more DATA ITEMS. Each DATA ITEM is constitutes a row in a table (Figure 29) and represents a possible data field with a set of rules for its extraction. The first column is the name of the field 131. The second is the 'location indicator 290.' The next five columns 291 comprise a bounding element as previously presented. The next column is the 'use prefix indicator,' 292 and the last, the default field value 293. A DATA ITEM is extracted and returned from the web page if a minimum of data fields are present and valid, as well as the resulting DATA ITEM itself.

The 'location indicator' tells the agent where to find the data field value.

There are several options:

- unused - this field is not in use for any DATA ITEMS to be created from this page.
- main - this field is collected from the Main Page, and to be included with every DATA ITEM.

- frame - this field is collected from the Frame Page, and is to be included with every DATA ITEM collected from this Frame Page only.

- sub - this field is collected from the Subpage, and included with the DATA ITEM collected from this Subpage.

- trans – this field is extracted from the Transition Page; this page is a physical web page that must be loaded by the agent in order to find the desired value for this field. The http address for this page is defined, for all Channels, to be the value extracted for the data field TR. Thus, if a Channel is to possibly use Transition Pages, then one of the data fields must be TR.

- build.

- default - rather than extract this field from the web page, use the value of the search parameter that this field is linked to if it is set, or use the default value from the list or type-in field present in the last column; if a list is present, it is constructed from the possible values for the linked input search parameter. The default specified in this latter situation takes precedence over the former if both exist.

- subform - the 'prefix indicator' is set to 'yes' or 'no.' 'Yes' indicates that after field value is constructed, the prefix indicator value as specified in the general specifications should be used as a prefix for the final value for this field.

The default field value is a list or type-in field, as previously explained. It is used only if the field location is set to default, otherwise it may be ignored.
A SEQUENCE is made up of a series of PAGES on a web site. As each PAGE has a PAGE DESCRIPTION, so too is there a SEQUENCE DESCRIPTION programmed for the SEQUENCE. The SEQUENCE DESCRIPTION is made up of a list of PAGE's, the navigation URLs for navigating to the first PAGE in the SEQUENCE and stepping through the remainder, and the SUBSTITUTION RULES for mapping one or more input parameter values into these URL's.

The series of PAGES that make up the SEQUENCE are specified from the panel shown in Figure 30. At first there is only an ‘insertion’ button 301 with the label “+0” on it. This and a ‘y’ numbered button 302 can be clicked to insert a new PAGE into the SEQUENCE at that point. The resulting control panel is shown in Figure 31. When a PAGE has been added, the SEQUENCE then shows this PAGE after the insertion button - in the example below a PAGE named “StartSearch” 310 has been added. Another insertion button appears after the page, to enable insertion of another PAGE - the button labeled “±1” 302 in the example below. Thus, the PAGE's can be added. Moving to Figure 31, as previously indicated, the control panel appears when a new page is added to the SEQUENCE. Any page that exists for any site in the Channel may be added. The page to be inserted 310 is highlighted from the menu of available page 312. Following insertion, following page insertion, the ‘return’ button 313, takes the user back to the sequence control panel of Figure 30.

The SEQUENCE starting URL is established via the control page shown in Figure 28. This is the standard ‘search URL’ form, described in detail previously as part of the PAGE DESCRIPTION's continuation page element. The difference here is that into the ‘bounding element’ values it is possible to embed special ‘substitution rule tags.’ These tags provide a mapping from input user query parameter values to elements of the final URL.

Figure 32 shows a control panel for specifying rules associated with each substitution rule tag that may be embedded in a search URL. The possible
tags are dynamically created as each input search parameter. They can be inserted anywhere in the search URL. When inserted, they must be surround by pairs of ‘less than’ and ‘greater than’ symbols. For example, the CY field would be embedded as <<CY>>. The complete set of rules for mapping the input search parameter to a value to use in the URL are listed below in the ‘Sequence URL substitution rules.’ In the first column, the search fields 320 are listed. All are candidates to be tags in the search URL. In the ‘Rule’ column 321 a list of possible rules to select from is displayed. The ‘Comment’ column 322 is for entering whatever comments the developer may wish. The wildcard 323 is a value to use if a substitution was requested, but the user failed to enter a value for the corresponding input search parameter. A list of substitution values, matched to input search parameter fields is given in the ‘Uselist’ column 324.

Data fields to be extracted come in one of two types; the most common being whatever value is retrieved is to be saved “as is”. But other fields are to be ascribed values, and these values are dependent upon the value extracted from the file. This is established by linking a search field to a data field as previously described. Then, when the SEQUENCE DESCRIPTION page is viewed, a special section under the MATCH RULES will appear for each data field, each possible. Pre-defined values for the data field will be listed, which are actually the possible input values for the search parameter. Then, the programmer needs to type in the value or values (separated by ‘II’) that, if the data field string extracted matches, indicates that this input search value should be returned for this data field.

An example is shown in Figure 33: the input parameter property type (TY) is linked to the output data field TY. Suppose the string ‘residential’ was typed into the ‘H’ field 330. Now suppose that in extracting a property from the web, for this site, the string found in extracting the data field TY was indeed ‘residential.’ Rather than return this string, an automatic substitution is performed, and ‘H’ returned instead. If no match is made, then the actual string found is returned despite it being a “linked field”.

-45-
The system currently provides a facility for testing a channel by executing a search in a special debug mode, shown in Figure 34. These features are only available from the main admin page and not visible to a consumer. An example of the search panel with these options visible is included on the right.

Debug mode can be set to ‘on’ or ‘off’ 340. Turning on will result in dozens of additional text messages dumped out at each step in the execution of the search. The system can also be directed to search a specific site in the channel 341, overriding all other site selection rules that might otherwise be invoked in forming the list of sites the agent will visit in response to a user-initiated query.

PARSING RULES

Position in file for an ELEMENT = bounding element
bounding element = text from START to END
START = position(start) + position(begin) + begin offset
END = position(end) + end offset
start, begin, end = TEXT Ref
begin, end = POSITION

Each data field is extracted via a bounding element specification as shown above gain. There are a host of possible parsing commands that can be embedded in each of these parameters.

start, begin, end - without any special commands just referenced as a string to step to in the current text from the current position

start only - if it starts with ll or + it is a special extraction rule, and begin, begin offset, end and end offset are ignored; the next character is the rule - currently the only rule is a t; the t rule indicates that the string is to be formed
as whatever comes between a pair of `<td>` tags in the page; the page is parsed case-insensitive until it finds the correct pair of such tags (see the argument described below), and the text in between extracted; all other html `<>` tag pairs and their contents are extracted and the resulting string returned

- there may be more than one `<td>` pair available; the argument that immediately follows the rule character specifies which of these pairs to use

- putting this all together, an example is: `||t5` which would indicate find the 5th pair of `<td></td>` in the current PAGE ELEMENT, and extract the string found between them according to the formula outlined above.

`start, begin, end` - «U indicates that the search URL that resulted in the current page be used

- sometimes a structural element is desired as a reference, but there are more than one and no easy way to get to the one desired; the way to achieve this is via the use of ! as the first character (repeater command), followed by a number 1-9 to indicate which element, followed by the element itself. For example, `!5<b>`, where the 5 indicates skip up to the 5th element, and the `<b>` is the element

- two choices are possible via use of the |OR| structure, e.g. `string1|OR|string2`; the page is searched for, either the string `string1` or `string2`; which ever comes first is used as the reference and the other ignored.

- if `begin` but no `end` specified, the `end` is assumed to be the next blank space after the `begin` position.
• if *end* but no *begin* specified, the *begin* is assumed to be the first blank space before the *end* position.

• the character ^ is replaced with a carriage return in any of these fields.

• the character * is replaced with a dollar sign ($) in any of these fields.

begin/end offset - usually a forward offset from the current position in number of characters, e.g. 5, 17, 23, etc.

• can also be a backwards reference; the first character is a minus sign (-), the next characters the offset, and the next the string to step back to; e.g. an offset could be -7</a> by which the intent would be to look backwards from the current position in the page to the first </a> tag it found, and then increment 7 positions forward from this; the resulting position in the file would be what the offset returned

SEQUENCE URL SUBSTITUTION RULES

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>don’t care parameter</td>
</tr>
<tr>
<td>LC</td>
<td>convert to lowercase</td>
</tr>
<tr>
<td>UC</td>
<td>convert to uppercase</td>
</tr>
<tr>
<td>X</td>
<td>extract string to use; requires uselist=ur</td>
</tr>
</tbody>
</table>
| R    | range match - a series of ranges are presented in the uselist; each range has a value associated; if the input
value falls within a range, then the corresponding value is returned; e.g. if the input value was 1000 for parameter, the rule was set to R, and the use list was 0-500=1, 501-1499=2, 1500-3000=3 then the value substituted would be 2, as the input value of 1000 fell into the 501-1499 range which had a value of 2 associated with it.

/value = input value/1000

STATE special - assumes input parameter is a U.S. state code (AL, KY, NY, etc.) and substitutes the full name of the state capitalized, e.g. ALABAMA, KENTUCKY, NEW YORK

state same, but all lowercase, e.g. alabama, kentucky, new york

State same, but capitalized for formal English, e.g. Alabama, Kentucky, New York).

Although the invention is described herein with reference to the preferred embodiment, one skilled in the art will readily appreciate that other applications may be substituted for those set forth herein without departing from the spirit and scope of the present invention. Accordingly, the invention should only be limited by the claims included below.
CLAIMS

What is claimed is:

1. A system for rapid development and deployment of object-oriented intelligent agent search and comparison applications, said system comprising a plurality of software modules, wherein said applications are built from one or more of said modules, said modules being assembled to form said applications; and

wherein said applications are easily developed, deployed and administered by individuals lacking programming skills through the use of an intuitive, graphical user interface; and

wherein said applications are readily adapted to any industry and any type of information.

2. The system of Claim 1, wherein said plurality of software modules includes:

a live agent coordinator, said live agent coordinator determining if and when one or more agents are to be invoked as part of an intelligent agent application execution, wherein said agent coordinator launches and monitors said invoked agents;

a network live pull agent, wherein said live pull agent queries a network resource in real time, in response to a user-generated request, and performs tasks involving information extraction and subsequent display or use of said extracted information;

a network archiving pull agent, wherein said archiving pull agent queries a network resource in real time, in response to a user-generated request, and performs tasks involving information extraction, and archives said extracted information for later use;

an archive-based search agent, wherein said archive-based agent searches and displays information previously extracted and archived by said archiving pull agent;
a network push agent, wherein said network push agent sends
information from a local site to a remote site via ftp or http;
a registration agent; wherein said registration agent can register a
user-generated query or other task on behalf of said user;
an agent stealth pack, wherein said stealth pack provides agents
with the capability to interact with remote sites unobtrusively so that impact
on remote site performance is minimized;
an archiving agent scheduler, wherein said scheduler schedules
essential behind-the-scenes activities using local system facilities;
an agent balancing system, wherein said agent balancing system
monitors and adjusts agent-based actions on host servers, monitors each
server's load, and adjusts application parameters as necessary;
a high-performance cache, wherein said cache is used to create a
temporary buffer of search results so that search time may be shortened.

3. The system of Claim 1, further comprising a plurality of components,
said components serving as building blocks for said modules, and wherein
each of said components gives a specific functionality to said modules.

4. The system of Claim 3, wherein said plurality of components
includes;
a web extraction rule set, said web extraction rule set comprising a
set of actions for extracting information from a remote site for specific data
fields, wherein an agent loads said rule set when visiting a site looking for
particular information;
a web navigation tool set, said web navigation rule set comprising a
set of actions for instructing an agent how to navigate a remote site;
an SQL library, said SQL library comprising a set of routines for
querying and otherwise interacting with commercial databases;
a data storage library, said data storage library comprising a set of
routines for storage of retrieved information;
a web site identification rule set, said web site identification rule set
comprising a set of parameters for describing a site with information of
interest;

a presentation library, said presentation library comprising a set of
functions for displaying information;

a runtime agent library, said runtime agent library comprising a set of
routines for performing live agent-based searches;

a database archive library, said database archive library comprising
a set of routines for archiving results of agent-based searches;

system monitor routines, said system monitor routines comprising a
set of routines for monitoring web servers, database servers, and agents;

agent performance methods, said agent performance methods
comprising a set of routines for configuring agent-based actions; and

web site health methods, said web site health methods comprising a
set of routines for monitoring web sites included in an application.

5. The system of Claim 3, further comprising one or more component
builders, said component builders comprising software facilities for building
new components and modifying existing ones, said component builders
providing a web based interface so that a simple, flexible, programmer-free
means of managing components is available.

6. The system of Claim 3, further comprising a plurality of objects, said
objects comprising the fundamental building blocks of said system,
wherein said objects are grouped according to intended function into one of
said components.

7. The system of Claim 6, wherein an application including a specific
component automatically inherits a newly created object and its properties
when the newly created object is included in said specific component.

8. The system of Claim 1, further comprising an application
administrator, said application administrator comprising a web-based
interface to a plurality of controls for administering all operations of an application.

9. The system of Claim 8, wherein said controls include one or more of:
cache management;
session management;
live agent operation;
archiving agent operation; and
a system monitor.

10. The system of Claim 1, further comprising an application extender, said application extender comprising a web based interface to application parameters and corresponding current values, wherein said interface is usable to add, modify or delete said parameter values.

11. The system of Claim 1, wherein said applications are platform independent, so that underlying software runs on any platform, and interfaces seamlessly with commercial database applications.

12. An architecture and system for an intelligent agent-based application capable of tracking dispersed, related web-based information on any subject and in any industry, said architecture comprising:
a proxy/router server in communication with a network;
an html server, said html server running one or more web server processes for serving static content, said html server in communication with said proxy/router server
a CGI server, said CGI server running one or more web server processes for serving any of dynamic and interactive content, said CGI server in communication with said proxy/router server;
an intelligent agent subsystem in communication with said proxy/router server; and
a database subsystem in communication with said proxy/router server;
wherein said proxy router server controls all of computers, processes and subsystems of said architecture and system.

13. The architecture and system of Claim 12, wherein said intelligent agent subsystem comprises:

1. one or more computers, said computers adapted to launch high-performance intelligent agent processes capable of performing information search, retrieval and comparison tasks associated with at least some application user services.

14. The architecture and system of Claim 13, wherein said proxy/router server comprises an intelligent agent traffic controller, and an intelligent search agent launcher.

15. The architecture and system of Claim 12, wherein said database subsystem comprises:

1. a database server, comprising a computer running one or more relational database server processes; and
2. a plurality of very large-scale relational databases wherein said database server is in communication with said databases.

16. The architecture and system of Claim 15, wherein said databases include an intelligent agent knowledge base database, a data storage database, and a users database.

17. The architecture and system of Claim 16, wherein said knowledge base database houses site and channel information, navigation rules, and parsing rules.

18. The architecture and system of Claim 16, wherein said data storage database houses a long term archive of retrieved information and short term caches.
19. The architecture and system of Claim 16, wherein said users database houses user information for registered users of said application.

20. The architecture and system of Claim 16, wherein said proxy/router server comprises a data portal to said database subsystem, wherein information is routed to and retrieved from said database through said database server.

21. The architecture and system of Claim 12, further comprising one or more client computers running web browser processes in communication with said network, wherein a user accesses said application.

22. The architecture and system of Claim 21, wherein a user accesses said application directly.

23. The architecture and system of Claim 21, wherein a user accesses said application via an affiliate server that links to said application.

24. The architecture and system of Claim 12, where users may be in one of the categories: random users, registered users and merchants, wherein said application provides an array of user services, and wherein portions of said user services are exclusive to specific categories and a portion is common to all categories.

25. The architecture and system of Claim 12, wherein said application provides an array of private, administrative services, and wherein said administrative services provide functions for creating, monitoring and manipulating public services, and for monitor and control of said system.

26. The architecture and system of Claim 12, wherein modes of operation for said application include:

   instant response mode, where information is immediately sought; and
delayed response mode, where a user-initiated query is saved and information found and retrieved later.

27. The architecture and system of Claim 26, wherein said instant response mode includes the options:
   live pull;
   archived pull; and
   live pull with cache lookup.

28. The architecture and system of Claim 26, wherein said delayed response mode includes the options:
   one-time live pull;
   continuous live pull;
   one-time archived pull;
   continuous archived pull;
   web push, where a user query is registered with remote sites and collected results are pushed back to said application on behalf of said user.

29. The architecture and system of Claim 28, wherein said collected results are displayed to the user when they log on.

30. The architecture and system of Claim 28, wherein said collected results are e-mailed to said user.

31. The architecture and system of Claim 12, further comprising a relational database structure, said database structure comprising a plurality of tables.

32. The architecture and system of Claim 31, wherein said tables include:
   channels, wherein a channel comprises a broad topic category of network information;
searches, wherein a search comprises a more specific information category and wherein channels are composed of searches; compares, comprising rules for comparing query results; defaults, comprising rules for filling default values for fields; and filters.

33. The architecture and system of Claim 12, wherein said architecture is scaleable.

34. The architecture and system of Claim 12, wherein said architecture is platform independent.

35. The architecture and system of Claim 12, wherein said system is automated, so that tasks associated with maintenance and operation of said application require minimal human intervention.

36. The architecture and system of Claim 12, further comprising a graphical user interface usable for creation, deployment and administration of user services, and for system administration and monitoring.

37. A method of customizing and administering an intelligent agent-based application capable of tracking dispersed, related web-based information on any subject and in any industry, comprising one or more of the steps of:

creating new channels, where a channel comprises a broad, top-level subject category;
modifying existing channels;
specifying one or more cache rules;
creating new topics, where a topic comprises a more specific subject area than a channel, and wherein a channel is composed of topics;
administering and modifying existing topics; and testing and debugging said application.
38. The method of Claim 37, wherein said channel creation step comprises the steps of:
   providing a user interface for creating said channel;
   accessing said interface;
   naming said channel;
   assigning said channel a key name;
   populating said channel with topics; and
   saving said channel.

39. The method of Claim 37, wherein said channel modification step comprises the steps of:
   providing a user interface for modifying said channel;
   accessing said interface;
   selecting a channel from a menu of existing channels;
   optionally, modifying said channel’s name value;
   optionally, modifying said channel’s key name value;
   optionally, modifying a selection of topics, wherein said channel is populated with said topics; and
   saving said channel.

40. The method of Claim 37, wherein said cache rule specifying step comprises the steps of:
   providing a user interface for specifying said cache rule;
   accessing said user interface;
   selecting a topic from a list of said topics;
   selecting one or more fields from a data item within said topic to be cached, wherein a key name for said item to be cached is formed by concatenating values of said selected fields;
   selecting a field from said item to be cached, wherein the current date is to be inserted;
   selecting fields from said item to be cached wherein a default value is specified for said selected fields; and
   saving said cache rule.
41. The method of Claim 37, wherein said topic creation step comprises the steps of:

   providing a user interface for creating said topic;
   accessing said user interface;
   specifying a name for said topic;
   specifying a key name for said topic;
   saving said topic; and
   activating said topic within one or more channels.

42. The method of Claim 37, wherein said topic administration step comprises one of more of the steps of:

   managing the cache for said topic;
   specifying agent actions;
   managing sessions;
   managing user search profiles;
   specifying search parameters;
   creating and modifying data fields;
   specifying archive and cache rules;
   specifying display rules;
   specifying web sites; and
   specifying rules for interaction of intelligent agents with web sites.

43. The method of Claim 42, wherein said cache management step comprises the steps of:

   providing a user interface for cache management;
   accessing said user interface;
   indicating a number of items in the cache;
   optionally, emptying the cache;
   specifying a maximum age for items in the cache;
   specifying a time interval for a system process to check the cache and remove items exceeding said maximum age; and
   saving cache settings.
44. The method of Claim 42, wherein said agent action specifying step comprises the steps of:
   providing a user interface for specifying agent actions;
   accessing said user interface;
   specifying a maximum wait time for initiation of an agent action;
   specifying a maximum number of retries for accessing a site that is busy or non-responsive;
   specifying a maximum wait time for a site to respond to a query;
   set ‘display presort’ on or off;
   specifying number of returned items per page per site to be displayed;
   specifying maximum number of returned items per page to be displayed and
   saving agent action settings.

45. The method of Claim 42, wherein said session management step comprises the steps of:
   providing a user interface for session management;
   accessing said user interface;
   indicating a current number of user sessions for a topic;
   optionally, removing all current sessions;
   specifying a maximum age for user sessions; and
   saving session management settings.

46. The method of Claim 42, wherein said user search profile management step comprises the steps of:
   providing a user interface for user search profile management;
   accessing said interface;
   indicating the number of profiles current profiles;
   optionally, removing current profiles from the system;
   executing a profile based on said profile’s assigned ID; and
   saving said profile settings.
47. The method of Claim 42, wherein said search parameter specification step comprises the steps of:
   providing an interface for creating and modifying search fields;
   accessing said interface;
   specifying a key name for said search field;
   specifying variable name;
   specifying a label;
   specifying a field type;
   specifying choices for multiple choice fields;
   specifying size and maximum string length for text entry fields;
   specifying linking field; and
   saving said settings.

48. The method of Claim 42, wherein said data field creation step comprises the steps of:
   providing a user interface for creating and modifying data fields;
   accessing said user interface;
   specifying a key name for said field;
   specifying a label;
   specifying a field type;
   specifying a maximum field size for text fields; and
   saving data field settings.

49. The method of Claim 42, wherein said archive and cache rule specification step comprises the steps of:
   providing a user interface for specifying said rules;
   accessing said user interface;
   selecting a channel from a list of said channels;
   selecting one or more fields from a data item within said channel to be cached, wherein a key name for said item to be cached is formed by concatenating values of said selected fields;
selecting a field from said item to be cached, wherein the current date is to be inserted;
selecting fields from said item to be cached wherein a default value is specified for said selected fields; and
saving said cache rule.

50. The method of Claim 42, wherein said display rule specification step comprises the steps of:
providing a user interface for specifying display elements;
accessing said user interface;
specifying a key name for said display element;
specifying a column width;
specifying a table column header label;
specifying an internal variable name;
specifying a link field;
specifying a field type, where said types include 'regular' and 'image';
and
saving display element settings.

51. The method of Claim 42, wherein said web site specification step comprises the steps of:
providing a user interface for specifying web sites;
accessing said user interface;
specifying a site code;
specifying a URL for said site;
specifying a name for said site;
specifying a site type, said types including web, Usenet, FTP;
specifying whether said site is active or inactive; and
saving said site settings.

52. The method of Claim 42, wherein said step of specifying rules for interaction of intelligent agents with web sites comprises the steps of:
selecting a site from a channel;
specifying one or more sequences for said site, wherein a sequence comprises the order in which pages from a site are to be navigated by said intelligent agent;
specifying one or more page descriptions for said site; and

53. The method of Claim 52, wherein said site selection step comprises the steps of:
providing a user interface for selecting a site;
accessing said user interface;
selecting a site from a list of sites on said channel; and
one of creating and modifying a search agent for said site.

54. The method of Claim 52, wherein said sequence specification step comprises the steps of:
providing a user interface for specifying sequence descriptions;
accessing said user interface;
specifying a list of pages and order of navigation for said intelligent agent;
specifying navigation URL’s for accessing a first page in said sequence and for stepping through remaining pages of said sequence; and
specifying substitution rules for mapping one or more input parameter values into said URL’s; and
saving said sequence description.

55. The method of Claim 52, wherein said page description specification step comprises the steps of:
providing a user interface for specification of page descriptions;
accessing said user interface;
defining general page properties;
defining page continuation properties;
defining page extraction rules; and
saving said page description.
56. The method of Claim 55, wherein said general page definition step comprises the steps of:
   defining framepages according to bounding elements of said framepages;
   defining subpages according to bounding elements of said subpages;
   listing data fields of said Subpages
   defining a page field prefix, where said page field prefix comprises a string appended to a data field, so that a complete URL for said data field is given; and
   saving said general page properties.

57. The method of Claim 56, wherein said intelligent agent parses HTML source code according to said bounding elements, and wherein a bounding element is defined by one or more components, said components including:
   value;
   start;
   begin;
   begin offset;
   end; and
   end offset.

58. The method of Claim 57, wherein said components: start, begin, and end constitute TEXT_REF’s.

59. The method of Claim 57, wherein said components: begin offset and end offset constitute are numerical values used to increment or decrement resulting positions found via said begin and end elements.

60. The method of Claim 56, wherein said bounding elements are referenced from a starting location of a current page element.
61. The method of Claim 55, wherein said page continuation definition step comprises the steps of:
   defining a continuation indicator according to bounding elements of said continuation page, wherein a continuation comprises a conceptual page spanning a plurality of physical pages;
   defining search URL's, wherein a search URL comprises a construct that results in a URL of a web page on a site being created, said page being loaded in at a future time by said intelligent agent; and
   saving said continuation information.

62. The method of Claim 61, wherein said continuation bounding elements comprise one or more components, said components including:
   value;
   start;
   begin;
   begin offset;
   end; and
   end offset.

63. The method of Claim 62, wherein said search URL bounding elements comprise one or more components, said components including:
   tag prefix;
   tag body; and
   tag suffix;
   for a page having an http method 'GET.'

64. The method of Claim 62, wherein said search URL bounding elements comprise one or more components, said components including:
   arg prefix;
   arg body; and
   arg suffix;
   for a page having an http method 'POST.'
65. The method of Claim 55, wherein said extraction rules definition step comprises the steps of:
   specifying a field name;
   specifying a location indicator, wherein said location indicator tells
   said intelligent agent where to locate value for said field;
   specifying bounding elements;
   setting a prefix indicator to 'on' or 'off'; and
   setting a default value for a list or type-in field.

66. The method of Claim 65, wherein said location indicator is one of:
   unused;
   main;
   frame
   sub
   trans;
   build;
   default; and
   subform.

67. The method of Claim 65, wherein said field bounding elements comprise one or more components, said components including;
   value;
   start;
   begin;
   begin offset;
   end; and
   end offset.

68. The method of Claim 37, wherein said testing and debugging step comprises the steps of:
   providing a user interface for testing and debugging;
   accessing said user interface;
   setting debug mode to 'on';
executing a search;
optionally, limiting said search to one specific site in a channel; and
optionally, launching said search in one of serial and parallel mode.
FIG. 2

AA registered user/member (my AA)

AA merchant (biz AA)

AA Public, User Services
FIG. 3
NewRealty's RateHunt

The Big Home Mortgage Rate Finder
Our Intelligent Agents Search Multiple Lender Websites To Save You Time!
Powered By

Ask Around!

Search:

Loan Type
- 30 Yr. Fixed
- 15 Yr. Fixed
- Fixed for 10 Yrs.*

Points from
- Please Select

Property's State
- Please Select

Purpose of loan
- Refinance a Home

Loan Amount

Estimated Market Value

Retrieve Existing Profile:
Your email: [Input Field]
Go

FIG. 6
**NewRealty's RATEHUNT**

The Big Home Mortgage Rate Finder
Our Intelligent Agents Search Multiple Lender Websites To Save You Time!
Powered By

Ask Around!

---

**Search:**

- **Loan Type:**
  - 30 Yr. Fixed
  - 15 Yr. Fixed
  - Fixed for 10 Yrs.
- **0 Points**

- **Property’s State:** California
- **Purpose of loan:** Cash
- **Loan Amount:** $400000
- **Estimated Market Value:** $700000

---

**Save this Profile**

- **My Email:**

**Send Me Updates**

- ○ daily
- ○ weekly
- ○ monthly

---

**Search**

- **0:00 time has elapsed**
- **80% of the search complete**
- **5 loans found so far**

---

**Results - Fri Nov 06 5:02 PM PST 1998**

<table>
<thead>
<tr>
<th>Product</th>
<th>Rate%</th>
<th>APR %</th>
<th>Points %</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>15YRFIXED</td>
<td>7.125</td>
<td>7.288</td>
<td>0.000</td>
<td>Countrywide Home Loans, Inc.</td>
</tr>
<tr>
<td>15YRFIXED</td>
<td>7.750</td>
<td>7.750</td>
<td>0.000</td>
<td>Bank of Boston</td>
</tr>
<tr>
<td>15YRFIXED</td>
<td>7.250</td>
<td>7.289</td>
<td>0.000</td>
<td>HomeShark</td>
</tr>
<tr>
<td>15YRFIXED</td>
<td>7.375</td>
<td>7.416</td>
<td>0.000</td>
<td>HomeShark</td>
</tr>
<tr>
<td>15YRFIXED</td>
<td>7.375</td>
<td>7.424</td>
<td>0.000</td>
<td>HomeShark</td>
</tr>
</tbody>
</table>
AskAround.com Administration

Shopping Online
* Auctions

Real Estate
* Real Estate Listings
  * Mortgages

Careers & Employment
* Resumes
  * Jobs

People
* Find Someone
* Personals & Matchmaking

Internet & Computers
* Multimedia
  * Web Casts

Fun & Entertainment

FIG. 8
Administration Menu

For this application, you need to specify [which Web Sites to search], [which Parameters are needed to search each site], [what Date Fields you expect to find and extract], [how the Data is to be Archived or Cached], and [how the Data Fields are to be Displayed].

For each of these sites, you need to specify [how to use the search parameters to find pages on the site], navigate through them, and extract and filter the desired data fields.

AskAround Real Estate Listings

AskAround System

Main Admin Page
Data Storage Rules: [Write to Cache] [Comparison] [Defaults] [Filters]
System Components: [Channels] [Searches]
Add or Update Real Estate Listings Cache Rule

Key: **RES**

**Build Keyname from:**
- AN -- X AO -- X BA -- □ BR -- □ CY -- □ DA -- □ LS -- □ ML --
- PR -- □ RM -- □ SA -- □ ST -- □ TH -- □ TR -- □ TY -- □ PS --

**Date Field:**
- □ AN -- □ AO -- □ BA -- □ BR -- □ CY -- □ DA -- □ LS -- □ ML --
- □ PR -- □ RM -- □ SA -- □ ST -- □ TH -- □ TR -- □ TY -- □ PS --

**Fields to Default:**
- □ AN -- □ AO -- □ BA -- □ BR -- □ CY -- □ DA -- □ LS -- □ ML --
- □ PR -- □ RM -- □ SA -- □ ST -- □ TH -- □ TR -- □ TY -- □ PS --

Add/Update this Rule  RES  ACT  RUM  Goto this Field

Delete Existing Cache Rule:  RES  ACT  RUM  □ Delete

**FIG. 13**
**FIG. 14**

- Current size: 16
- Empty Cache Now
- Check cache every 6 hours
- Remove items 32 hours old
- Cache pre-archiver: Off On
- Archive site: YAH CEN

Save Cache Settings

**FIG. 15**

- Wait at most 60 secs for a user search
- Try a site a max of 3 per search
- Wait a max of 5 secs for a site to respond
- Display presort: Off On
- Max# per site per page to display
- Max# per page to display

Save Agent Actions Settings
FIG. 16

Sessions
- Current # sessions: 3 Remove All Sessions Now
- Check sessions every 1 hour
- Remove sessions 10 minutes old

Save Session Settings

FIG. 17

Search Profiles
- # Profiles in system: 0 Remove All Profiles Now
- Execute existing profile:

Go
Add or Update Ask Around Real Estate Listings Site Search Field

Key: CY
Descrip: City
Argname: city

Link to Data Field? --None--

Loop On this Field? --No O --Yes O

Add/Update this New Field
Delete this Field

FIG. 18
Add or Update AskAround Real Estate Listing Data Display Element

Key: 12
Width: 75

Label: Price
DisplayField: PR
DisplayField Type: Regular

Sort name: price
Link Field: TR

Add/Update this New Display Field
Delete this Field

FIG. 20
AskAround Real Estate Listings

Enter the Site Information, then Hit Save

|------------------------|-----|-----------|-------------------------------------------------------------|

| Preference? | ☑   |

<table>
<thead>
<tr>
<th>URL:</th>
<th><a href="http://homesearch.us-digital.com">http://homesearch.us-digital.com</a></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Site Name:</th>
<th>U.S. Digital Home WEB</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Type:</th>
<th>Web</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Status:</th>
<th>○ Active ○ Inactive (0 empty searches, last one was profile #692827)</th>
</tr>
</thead>
</table>

| Warn after how many empty trips? | 5 |

Add/Update this Site  Clear all Fields  Return (No Save)
IA Agent for Channel X launches a search

Site A visited

Site B visited

Site C NOT visited

Site D visited

Sites, Pages and Sequences available for AA Channel X

Sequence 1

Sequence 2

Sequence 1

Sequence 2

Sequence 1

Sequence 2

FIG. 24
FIG. 26
**FIG. 27**
Continuation Possible? ☐ YES ☐ NO

Indicator

Value: NEXT</a>
Begin: [ ] Offset: [ ] End: [ ] Offset: [ ]

URL to Continue

Access method: ☐ GET ☐ POST Operational method: ☐ Live/Archived ☐ Re

Tag Prefix Value: http://search.us-digital.com/hw/sear Start: [ ] Offset: [ ] End: [ ] Offset: [ ]

Tag Body Value: [ ] Start: [ ] Offset: [ ] End: [ ] Offset: [ ]

Tag Suffix Value: [ ] Start: [ ] Offset: [ ] End: [ ] Offset: [ ]

Arg Prefix Value: NEXT Start: [ ] Offset: [ ]
Begin: [ ] Offset: [ ] End: [ ] Offset: [ ]

Arg Body Value: [ ] Start: [ ] Offset: [ ] End: [ ] Offset: [ ]

Arg Suffix Value: [ ] Start: [ ] Offset: [ ] End: [ ] Offset: [ ]

Save Continuation Info

FIG. 28
Click on a "+[num]" button to insert a page into the sequence, or click on a page to remove it from the sequence.

FIG. 30

RHC Search Agent: Insert Page into Sequence

Existing Pages:
C21_StartPage
RHC_StartSearch
ELR_StartPage

FIG. 31
MATCH RULES

Data Field TY linked to Search Field TY

H: RESIDENTIAL
M: 
L: 
F: 

Data Field PS linked to Search Field PS

A: 
C: 
F: 
R: 

Update Match Rules

FIG. 33
<table>
<thead>
<tr>
<th><strong>Property</strong></th>
<th><strong>Value</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>City</td>
<td></td>
</tr>
<tr>
<td>State</td>
<td>None</td>
</tr>
<tr>
<td>Property Type</td>
<td>Home/Residential</td>
</tr>
<tr>
<td>Minimum Bedrooms</td>
<td>0</td>
</tr>
<tr>
<td>Status</td>
<td>For Sale By Agent</td>
</tr>
<tr>
<td>Minimum Price</td>
<td></td>
</tr>
<tr>
<td>Maximum Price</td>
<td></td>
</tr>
<tr>
<td>Minimum Bathrooms</td>
<td>0</td>
</tr>
</tbody>
</table>

**Debug Mode:**  [ ] No  [ ] Yes

**Include Only Site:** All Sites

**Live Method:**  [ ] Serial  [ ] Parallel

---

**Help & Tips**

**FIG. 34**