World Multiconference on Systemics, Cybernetics and Informatics

July 22-25, 2001
Orlando, Florida, USA

PROCEEDINGS

Volume I

Information Systems Development

Organized by IIIS
International
Institute of
Informatics
and Systemics

Member of the International
Federation of Systems Research

IFSR

Co-organized by IEEE Computer Society
(Chapter: Venezuela)

EDITORS
Nagib Callaos
Ivan Nunes da Silva
Jorge Molero

MedFerret: Client-Based Semantic Query Integrator*

Naphtali D. RISHE, Oksana DYGANOVA, Andriy SELIVONENKO,
Maxim CHEKMASOV, Alejandro MENDOZA
High Performance Database Research Center
NASA Regional Applications Center
Florida International University
Miami, FL 33199, USA

ABSTRACT

MedFerret is a semantic query merging application focused on dynamic data retrieval and on merging data available from medical sites on the Internet. MedFerret's system architecture is based on a heterogeneous database approach that utilizes the Semantic Object-oriented Data Model. The system is implemented as a Java applet on the client's computer and is, in fact, a multi-threaded Java web server that runs search queries against medical sites on the Internet and that returns results to several users simultaneously.

Keywords: heterogeneous databases, semantic schema, dynamic data retrieval, medical databases, Internet searching

1. INTRODUCTION

As no search service can be universally efficient or complete, the appearance of a new type of web-oriented software, often called meta-searchers has been triggered. A meta-searcher is an information retrieval system that is based on several web data sources distributed across the Internet. The web data sources are often highly autonomous, but provide similar information to the end user. An individual database is usually maintained behind the front page of the web site presented to the user. A web application integrating information from several web sites has to deal with the heterogeneity caused by the different data models and different query interfaces implemented in these underlying databases. We also need to take into account that some of web data sources have no data models at all but only a loose mapping between a web page form, e.g. a post-data input form, and html output.

In order to construct a web application that overcomes these obstacles, a global data model is needed to integrate the semantics of the data sources available via the WWW. One of the models is a heterogeneous database (HDB) approach that is under development at the High Performance Database Research Center (HPDRC). [1] Under this HDB system, users can access a number of databases with the extended system as if there were only one semantic database since we use the Semantic Object-oriented Data Model to construct a global data schema. The query interfaces of the HDB system are exactly the same as those provided by the Semantic Object-Oriented Database that is under development at HPDRC, see [2] for more details.

MedFerret is an example of an Internet application that illustrates some approaches to HDB system implementation in the real world. It is a fully functional in prototype mode and can be visited on the web. [3]

2. GENERAL APPROACH

From the user's perspective, MedFerret is a system that consolidates medical information the user is interested in into one form convenient for browsing and review. It is easy to use and has a number of advantages when compared to a conventional search engine. The conventional search engine allows the user to pose a simple matching query. It then returns a list of links to *static* web pages, which contain the words entered by the user in the search box. In many cases the link is accompanied by a portion of the source HTML page, which gives the user a flavor of the information she can get by visiting the link. On a similar request, MedFerret returns medical information that is *dynamically* retrieved from the web data sources' databases. The data received from the databases is usually better structured, well maintained and up-to-date. Eventually this data could contain complex data elements like images or other binary objects that make the resulting set more informative to the user.

From the HDB system perspective, we can look at MedFerret as follows. Consider a set of the Internet web sites, which are directly or indirectly related to medicine. We can call this set of web sites the 'medical Internet' or the 'Internet sub-world related to medical issues.' Of course, the medical Internet is not sufficiently structured, well-organized, or maintained to be considered as one

* This research was supported in part by NASA (under grants NAG5-9478, NAGW-4080, NAG5-5095, NAS5-97222, and NAG5-6830) and NSF (CDA-9711582, IRI-9409661, HRD-9707076, and ANI-9876409).

large medical database. We intend to focus on a set of web sites within the medical Internet chosen by certain criteria like medical topics covered, volumes of medical data, quality of medical reviews, etc.

We thoroughly study the semantics of each chosen web site. By doing so, we get a formal description of all the data elements available from the web site and the relationships between the data elements. This formal description can be expressed in a semantic schema [2], which we will call a semantic schema of the web data source. We describe the semantics of the web site in terms of the HDB system and the HDB system can then treat a set of medical web sites as a collection of heterogeneous distributed data sources.

The HDB system supporting MedFerret is based on a set of medical web sites. These data sources are well structured and organized and can be accessed by the user through the formal query mechanisms of the HDB system. In our current prototype version, however, MedFerret does not exploit all the functions of the HDB system. For example, the user is not allowed to send arbitrary ad-hoc queries to the data sources based on their semantic schemas. At this time the user is only allowed to enter search keywords to get the desired results.

3. MEDFERRET FEATURES

We are now ready to review the features of Medferret. When the user visits the MedFerret web site, the system tries to establish a web service on the user's computer. The web service is implemented as a Java applet. Before sending the Java applet to the user's side, the MedFerret server needs to check the user's authorization to run the web service. This is done by checking a username and password entered by the user. If the authorization process fails, the system is unable to establish the web service on the user's side. The MedFerret system also checks the current browser settings on the user's computer. The permissions should be set to allow a Java applet to be executed on the user's computer. If the browser security settings related to Java applets are not set appropriately, the MedFerret system is unable to start the web service.

When both steps of authorization and Java applet launch are successful, the web service capable of executing queries against the medical web sites is established on the user's computer. The web service will be active and listening for requests until the user closes the browser or leaves the MedFerret web page.

An important detail of this technique is that when the web service is active on the user's computer, it can be accessed by other external users to receive MedFerret information retrieval services. Applications are also able to use and retrieve medical data through the MedFerret web service. A typical example is when the user sends a request from Microsoft Excel to the MedFerret web service and the results are automatically downloaded into the Excel spreadsheet. Another example is when the user's program sends a request through a URL to the web service, which executes the query and sends the results back to the user's program. Thus the MedFerret web service on the client side emulates the MedFerret server on the user's computer.

The search capabilities are limited in the prototype version of MedFerret. Currently the user or her applications are able to submit only a set of keywords and retrieve data in a standard format. In the next versions, a number of additional parameters will be introduced to the search function, giving the user more freedom to specify queries to the medical web sites.

When the input parameters (keywords in the first version) are supplied, the MedFerret web service invokes a set of agent programs (agents) that collect data from the Internet. The agents are run in parallel to increase the performance of the system. Each agent has knowledge of the semantic schema of the data elements and their relationships for one particular medical web site. Thus the number of agents launched equals the number of medical databases queried via the Internet.

The agents return data to the MedFerret web service on the user's computer, which then merges these datasets into one large dataset and returns the resulting dataset to the user or her application. At present, the web sites chosen as data providers for the MedFerret service return data in a similar format that allows simple merging of the results. With an increased number of medical data sources, the results will become more diverse and the merging capability of the web service will need to be adjusted accordingly.

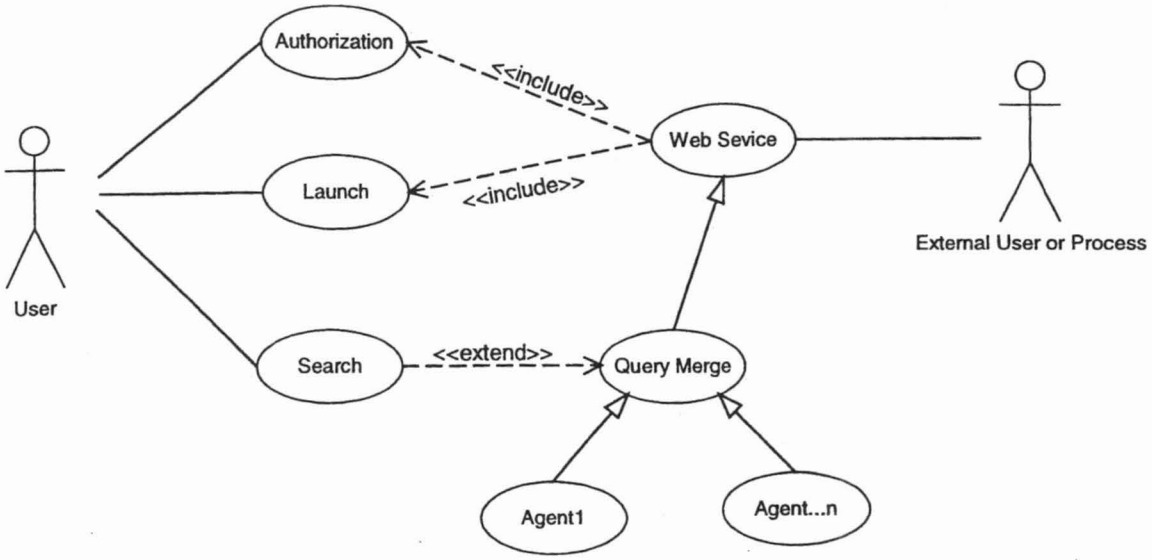
Diagram 1 portrays a case study of the described processes.

4. REQUESTS AND DATA FLOW IN THE MEDFERRET SYSTEM

Several key elements are involved in the processing of a user's request by the MedFerret system:

- Browser: The browser is installed and running to access the MedFerret web site.
- Web Service: The MedFerret web service is delivered and launched in the user's browser as a Java applet upon successful authorization and with appropriate Java applet support from the browser.
- Agents: The agents are run to access medical web sites, to send the requests to the web sites' medical databases, and to deliver the results.

Diagram1: MedFerret Use Case Diagram



- External Web Server CGI: This program usually serves as an intermediary between user requests and the medical database on each data source web site.
- Database: Each medical database is maintained and supported by an information service provider.

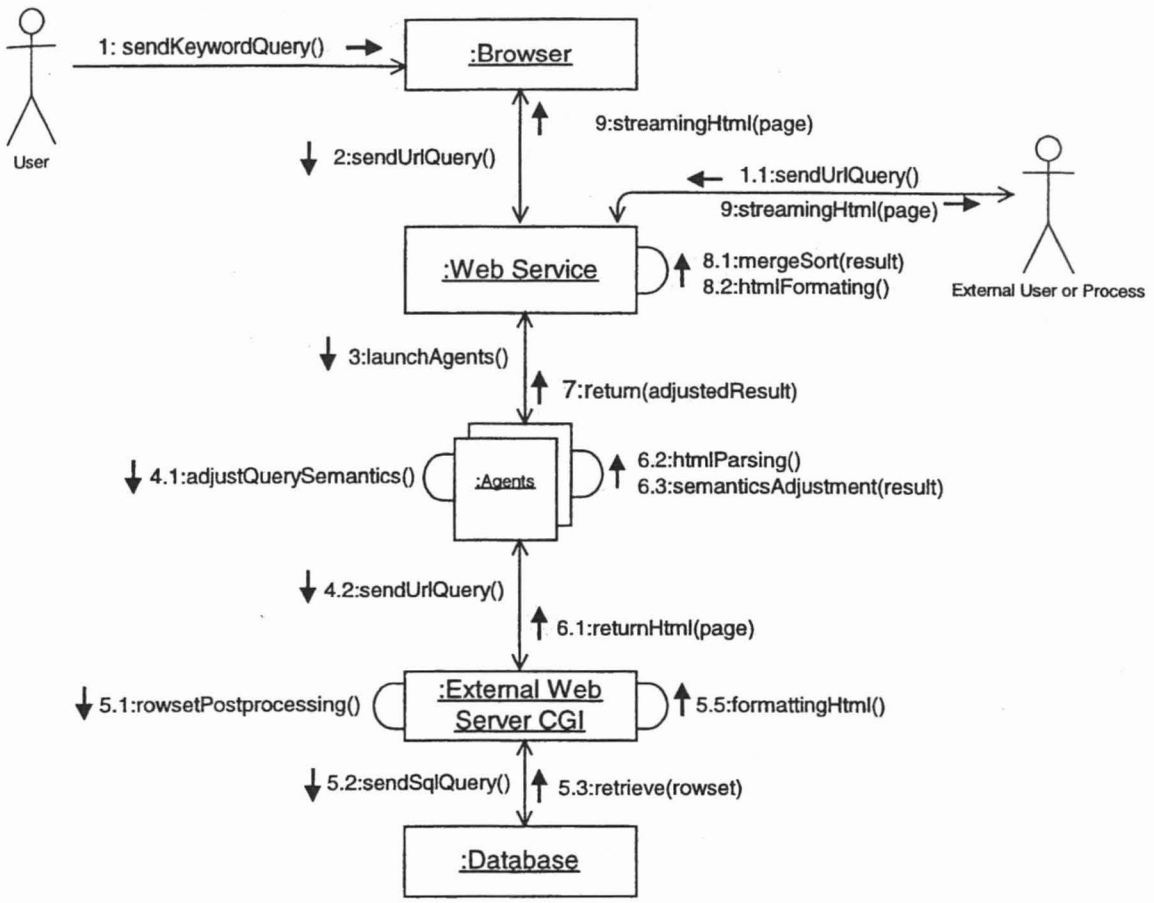
The process of sending queries and receiving results can be described as follows. The user types in and sends a set of keywords to the Browser, which will be used as search criteria. The Browser transforms the user's request into a URL and sends the URL to the Web Service. Requests to the Web Service may also come directly from external users or processes in URL form, thus avoiding the Browser. The Web Service launches a set of Agents. Each Agent adjusts the user's query semantics to the needs of its particular web site and sends requests to the External Web Server CGI of its assigned web site in URL form. The External Web server CGI transforms the input URL into a query that is sent to its site's Database for execution. The result returned by the database is often post-processed by the External Web server CGI, which formats the result into HTML and then returns the HTML page to the Agent, which, in turn, parses the HTML and adjusts its semantics to fit into the aggregated result that is to be returned to the user. It then returns the adjusted result to the Web Service, which compiles the aggregate result from all of the results returned by the Agents. Finally, the Web Service sorts the aggregated result set, formats it into HTML, and streams it back to the Browser or directly to external the user as HTML. Diagram 2 depicts the MedFerret collaboration process.

5. IMPLEMENTATION AND RESULTS

The prototype version of MedFerret was created using a Java 1.1 compatible applet and JavaScript. When a user comes to the MedFerret.com site, the user's browser loads an HTML frameset. This frameset consists of an upper narrow frame and a lower frame. JavaScript in the upper frame detects the version and brand of browser (Netscape or Microsoft IE) and loads the appropriate Java archive. The lower frame shows a "Please wait while application loading" message while this process takes place. When the Java archive is loaded into the upper frame, the browser verifies the archive's integrity and checks the validity of the digital signature. Inside the Java archive there are statements that request additional security clearances: "Writing data from a network connection," "Reading data from a network connection," and "Contacting with other computer over a network." The user is asked to grant these clearances based on the fact that the digital certificate indicates that a valid digital certificate, which was granted to Florida International University, was used to sign this application.

If the user does not grant such privileges, the MedFerret applet cannot connect to external web sites and cannot execute queries against them. If such privileges are granted, the Java applet starts its execution and calls JavaScript functions inside the HTML frameset via LiveConnect technology. This JavaScript function

Diagram2: MedFerret Collaboration Diagram



replaces the message "Please wait while application loading" in the lower frame with a MedFerret starting query page. This page has an HTML form for the user to type in a keyword query, as well as keyword query help and examples of query usage. At this time, MedFerret is fully functional and ready to accept user's queries. MedFerret intercepts TCP port 7000 on the user's computer and waits for queries, which should be sent via http protocol. When a user types a keyword query in the lower frame and presses the "Go" button, the target of HTML frame is http://localhost:7000/?search. The browser then sends out a query against the Web server, which is located on IP address localhost and on the port 7000. Assume that user had typed query:

women and "chest pain"

The query will be transformed by the browser into the URL:

http://localhost:7000/?search=women+and+%22chest+pain%22

where spaces were replaced by + signs and quotes by %22 expressions according to URL-encoding convention. MedFerret is a mutli-threaded Java Web Server, which launches a query thread on each http request. There could be more than one simultaneously running search query, since the user can open more than one browser window. The user can also launch search queries from other applications, such as Microsoft Excel or Microsoft Word. Queries can also be launched from other computers. In such cases, the URL http://localhost:7000/?search=keywords... should be replaced with the fully qualified domain name or IP address of the computer that is presently running an instance of MedFerret:

http://user.domain.com:7000/?search=keywords... or http://18.29.1.34:7000/?search=keywords...

MedFerret could thus effectively support external queries.

MedFerret is built using a "client-server" model and pure Java. The uniqueness of the MedFerret approach is the fact that both the client and the server could reside in the same browser. The following are some reasons why this approach is beneficial to the MedFerret application:

- MedFerret can serve queries that come from the local computer and from external computers.
- MedFerret can serve queries that come from the browser and from other applications (Word, Excel, database applications, etc.).
- MedFerret uses native browser code to render result text on the screen. In most cases this is more effective, precise and readable than such text rendered by Java. If MedFerret were to write textual output by means of a Java graphic library, its rendering speed and quality would usually be significantly worse than that of the browser using its own code.

MedFerret supports a direct URL invocation mode even if there is no running instance of MedFerret in any browser on the user's computer. This would be beneficial if MedFerret needs to be spawned from another web site with a pre-specified query. MedFerret could also launch a query instantly when invoked from a browser's bookmark.

6. CONCLUSION

We presented MedFerret as a semantic query merging application focused on dynamic data retrieval from the medical Internet. MedFerret's system architecture is based on an HDB approach. The same approach can be used to construct web applications related to other subject areas. For each potential application, the semantics of the corresponding web data sources should be captured and implemented in the agents.

The current implementation of the MedFerret system can be enriched by new functions. Additional means for specifying more complex queries to the medical Internet may be provided to user. The semantic schema of the medical Internet visited by the agents may be displayed. Other medical web sites may be studied and the corresponding agents implemented to broaden the search for information.

7. REFERENCES

[1] R.Athauda Integration and Querying of Heterogeneous, Autonomous, Distributed Database Systems, PhD dissertation, Florida International University, Miami, July 5, 2000.

[2] N.Rishe Database Design: The Semantic Modeling Approach. McGraw-Hill,Inc. N.Y., N.Y, 1992.

[3] Simultaneously Search Many Medical Databases: http://www.MedFerret.com

ISBN: 980-07-7541-2