# A GML-Based Open Architecture for Building a Geographical Information Search Engine Over the Internet

Jianting Zhang Le Gruenwald
The University of Oklahoma, School of Computer Science, Norman, OK, 73019
Contact author email: ggruenwald@ou.edu, Phone: 1-405-325-3498

## Abstract

This paper proposes an open architecture for building a geographical information search engine over the Internet based on the Geographical Markup Language (GML). It recognizes the autonomous and heterogeneous nature of the current practice of the Internet by adding GML-enabled wrappers for distributed servers and a mediator tier. The paper then presents a prototype based on the proposed open architecture using the University of Oklahoma's student addresses.

## 1. Introduction

Today's Web contains a great deal of information that can be geo-referenced. The National Academy of Sciences estimates that 80 percents of the information on the Web have a spatial component ([HREF 1]). These include coordination information, such as latitude/longitude and their various kinds of projections, mailing addresses that can be geocoded, and relative distance/direction information.

With the widely applications of personal portable communication and computation devices, such as cell phones, GPS devices, PDAs, and Palms, geo-referenced information (GRI) and applications are undergoing significant changes. The volume of personal GRI, such as location and moving direction, produced by GPS devices and cell phones, has a much greater volume than traditional GRI, such as street map and cadastral information. It is highly unlikely that this kind of personal GRI can be collected and managed by a centralized system. Instead, it is often embedded in various forms of documents, such as HTML pages, in an autonomous and distributed fashion. We believe that a geographical searching engine over the Internet, which is an extension of the current text-based search engines, such as Google ([HREF 2]) and Altavista ([HREF 3]), is highly desirable.

In this paper, we propose an open architecture based on Geographical Markup Language (GML) for building a geographical information search engine over the Internet. The architecture consists of three tiers: Distributed Servers, Mediator and Clients. A Distributed Server has software agent(s) to roam over the Web sites in the designated categories, parses Web pages to retrieve GRI and stores them in its local database. A Distributed Server is also responsible for wrapping GRI into a GML document when being requested from either a Client or the Mediator. The Mediator accepts requests from a Client and breaks the query into sub-queries and sends them to the corresponding Distributed Servers. Through a negotiation mechanism, the Mediator either accepts the query results from a Distributed Server in the GML format, or retrieves data set periodically from the Distributed Server and performs spatial queries on its own database on behalf of the Distributed Server. The Mediator wraps the query results into a GML document and sends it back to the Client. The Client accepts the user's queries interactively. It communicates with the Mediator or the Distributed Servers directly and renders the query results graphically.

The rest of this paper is organized as follows. Section 2 provides an overview of related work. Section 3 presents the proposed architecture. Section 4 describes a prototype implementation of the proposed architecture using the University of Oklahoma's student addresses. Section 5 reports the primary experiment results of the constructed prototype system. Finally Section 6 presents future work directions.

## 2. Related Work

The Internet mapping of geographical data, such as MapQuest.com ([HREF 4]) and Excite.com ([HREF 5]), has provided GRI access through the Internet and is increasingly gaining popularity. Major

Geographical Information System (GIS) vendors, such as ESRI ([HREF 6]), MapInfo ([HREF 7]), Intergraph ([HREF 8]) and AutoDesk ([HREF 9]), provide software components to publish geographical data online. An overview of Internet Mapping is presented in ([3], [10]). Usually the GRI in such systems is searchable either by keywords or positions. However, data as well as software in such systems are proprietary and completely controlled by vendors. In addition, the generated maps usually either in image formats (GIF/JPEG) or embedded objects in browsers which make it very difficult, if not impossible, for users to integrate search results from different systems or with other applications. The best-known academic project in the Internet mapping field is Alexandria Digital Library Project ([HREF 10]) funded by the NSF digital library program. In this project, collections of geographically referenced materials and services for accessing those collections are developed. A Gazetteer Server with a GUI is also built to map a place name onto an image. However, all the place information is static and does not associate with any Web pages. In general, the current Internet mapping technologies are mostly for GRI publishing in a centralized manner and not for gathering GRI over the Internet which is autonomous and distributed in nature.

Location-based Services (LBS) is an integrated technology of telecommunication and GIS. The key idea of LBS is that a portable device sends its location information to a gateway, the gateway search through its database to find the most relevant information near the location and sends it back to the client for further use. A problem with this technology is that the gateway must maintain a centralized GRI database to support queries. For example, OpenWave system ([HREF 8]), a product from Phone.com, requires every website to report its geographical location to its gateway in order for the website to be accessed by its mobile devices. This solution requires a well-established GRI infrastructure while anything not registered in the central database will not be accessible to end users. A more formal and open solution that specifically focuses on GRI infrastructure is proposed by SRI International ([HREF 1]). They proposed a new Top-Level Domain Name (TLD) named ".geo". According to their suggestions, the earth is divided into 1 by 1 degree cells and each cell has one or more GeoRegistries. All URLs register their geographical locations with these GeoRegistries and the GeoRegistries provide services to all GRI requests. However, the proposal is recently rejected by The Internet Corporation for Assigned Names and Numbers (ICANN, [HREF 12]) ([HREF 13]). From a practical perspective, one major shortcoming of the ".geo" TLD proposal is that it does not address the issue of how to deal with text-based GRI such as mailing address, which is more relevant to and widely used in our everyday life. Again, anything not registered in the central database will not be accessible to the end-user. In general, although LBS/TLD technologies provide mechanisms to gather and distribute dynamic GRI in real time, they still adopt a centralized architecture, which is not suitable for gathering geographical information from autonomous and distributed websites.

There has been a large volume of research in document classification ([5], [8]) and building search engines ([4], [6]). In practice, several popular search engines roam over the Internet to collect information and make it searchable. However, most of them only retrieve text information for keywords matching and do not support searching geographic information. Google attempts to support finding maps based on given addresses ([HREF 14]); however, it has no built-in mechanism to interpret addresses. What is does, basically, is still one-to-one word matching. For example, if we search by "1307 George Ave, Norman, OK, 73072", it will match both phone numbers and addresses that have "1307 " in them. Most searched results are irrelevant to what the user wants because word matching does not take any semantics or geographical context into consideration.

A bridge between text-based GRI (such as mailing addresses) to coordination based GRI (such as latitudes/longitudes) is a technology called Geocoding or Address Matching ([2]). Geocoding is essentially a special natural language processing (NLP) problem. Although currently there are a few commercial software products available ([HREF 15], [HREF 16]), few research papers focus on this topic. Recently, the Open GIS Consortium (OGC, [HREF 17]) has unofficially released Geocoder Service Specification ([HREF 18]) and is expecting implementation compliances in the near future.

To manage geographical information that is essentially a special type of spatial information, spatial data access and analysis methods are needed. Spatial index methods, such as Quad-tree and R-trees, are well studied in academia ([7]). However, few mainstream commercial database software products support spatial data access. Oracle Spatial Option is one of them that supports Quad-Tree/R-tree indexing methods ([HREF 19]). Spatial analysis methods, such as distance/direction analysis, buffer analysis, topological analysis, network analysis, and terrain analysis, are well studied in the GIS field and implemented in many commercial GIS software

packages. However, these software packages are usually standalone and do not provide a good integration mechanism with mainstream database software. Its major problem is that different GIS software vendors have different geographical data formats that result in poor interoperability.

XML-based data integration architectures ([1]) are becoming more and more popular since XML is a kind of text-based protocol that is easily processed and exchanged between users. GML, which is an extension of XML, is proposed by OGC to solve the GRI interoperability problem ([HREF 20]). Several spatial data types, such as point, polyline and polygon, as well as earth projection types are defined in GML DTD. Any software that supports GML can use geographical data in a GML document. An XML-based spatial data mediation infrastructure for global interoperability study is conducted in San Diego Supercomputer Center ([9]). Based on GeoXML, which is similar to GML but using a smaller DTD set, we also proposed and implemented a wrapper/mediator architecture to visualize integrated geographical information ([10]). However, data used in both studies is pre-collected and does not make use of rich GRI over the Internet. In contrast, in this study, we build software agents to roam over the Internet to collect geo-referenced information.

## Proposed System Architecture

The overall architecture of the system is shown in Fig. 1. The system adopts a three-tier architecture: the first tier is a distributed server. A distributed server has three components: a software agent, a local database and a wrapper. The software agent roams over all the websites being assigned according to a predefined schedule, retrieves all the GRI and their associated text information in the Web pages and then stores them in its local database. Since the majority GRI involved is of point type, the database could be relational database, object-oriented database or object-relational database. If the database supports geographical information accesses and queries, it should register this information in the mediator that we will describe below in detail. The wrapper extracts the relevant data from the local database and transforms it into the GML format when a valid request is received.

The second tier is the Mediator. Whenever the mediator receives a request from a client, it first determines which distributed server(s) is involved and breaks the query into sub-queries. It then finds the corresponding distributed servers in its meta-database. If the corresponding distributed server supports spatial data access and query, the Mediator then just sends a sub-query to the distributed server. If not, the Mediator checks the most recently data received from that distributed server. If the time span is within tolerance, the Mediator queries the duplicated data on that distributed server's behalf in its own database; otherwise it updates the duplication first before query. The mediator then sends the integrated query results also in the GML format back to the requested client.

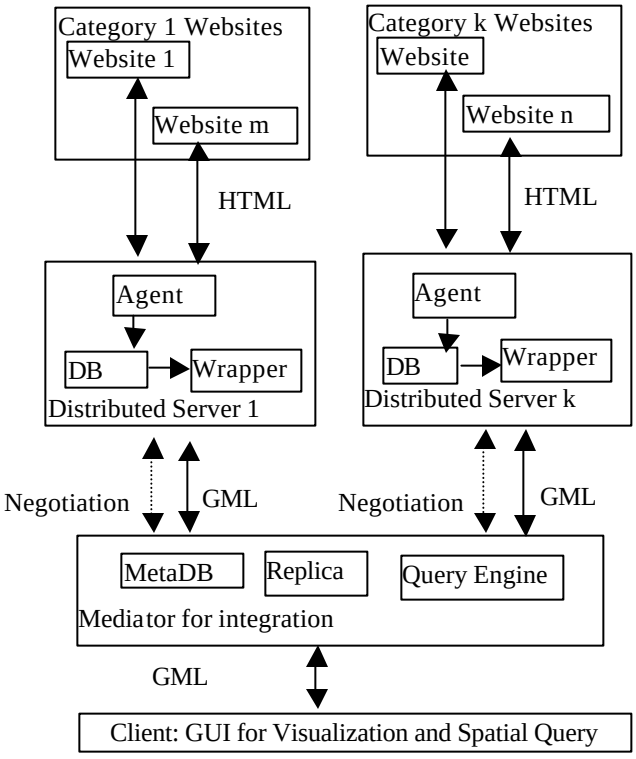


Figure 1. Proposed System Architecture

The third tier is the Client side. It supports geographical data visualization based on spatial queries. The client side communicates with the Mediator in real time and retrieve GML documents from the Mediator. The user can use Zoom In and Zoom Out to focus on different levels of details. Besides visualization functionalities, textual information associated with geographical locations is also displayed in the Client's browser and hyperlinks can lead the user to corresponding web pages.

Note that we use GML as the common communication protocol between the Distributed Servers and Mediator as well as between the Mediator and the Client. We have implemented a GML wrapper for flat file in our previous study ([10]). To make our work more general, in this paper we

implement GML wrappers for relational databases and object-relational databases.

## A Prototype Implementation

### *Motivation of the prototype*

Our first motivation in building the prototype is from the student contact information searching system at the University of Oklahoma with which the authors are associated ([HREF 21]). Given a full or partial student name, the system is able to return the full student name, email, status, college, street address, city, state, zip code and phone number. However, the system does not allow a user to perform queries on spatial relationships like "Search all the freshmen that are less than 5 miles away from me" or "List all the graduate students who live around the Parkview apartments". Our prototype system first retrieves student names from the university list at http://students.ou.edu and then sends the retrieved student names to the university's search engine ([HREF 21]) to retrieve the student information and stores it in our databases. The addresses are then geocoded using a commercial geocoding product from MapInfo ([HREF 7]). GML wrappers, mediator as well as client side visualization tool are built to demonstrate our ideas which will be explained in detail in the following sub-sections.

For demonstration purposes, we build two distributed servers to retrieve the information of students whose last names begin with "A" and "B", respectively. In our prototype system, one distributed server uses Microsoft SQL Server 2000 and the other distributed server uses Oracle 8.1.7. The mediator uses Oracle 8.1.7 to provide spatial query and analysis functions. All the modules, including software agents, wrappers, mediator and client component in our prototype system are written in Java.

### *Software agents for collecting geographical Information over the Internet*

The procedure is as follows: the software agent of the distributed server first sends a request to the designated websites. Then it parses the returned HTML pages into trees using html parser classes in SWING provided in JDK Enterprises Edition v1.3 from SUN ([HREF 22]). The agent then tries to find the student address information based on both the retrieved HTML page structure (such as the hierarchy of HTML tags) and keywords (such as "Address"). The addresses are then geocoded into longitude/latitude and stored in the corresponding databases in the distributed servers. The general process is shown in Fig. 2.

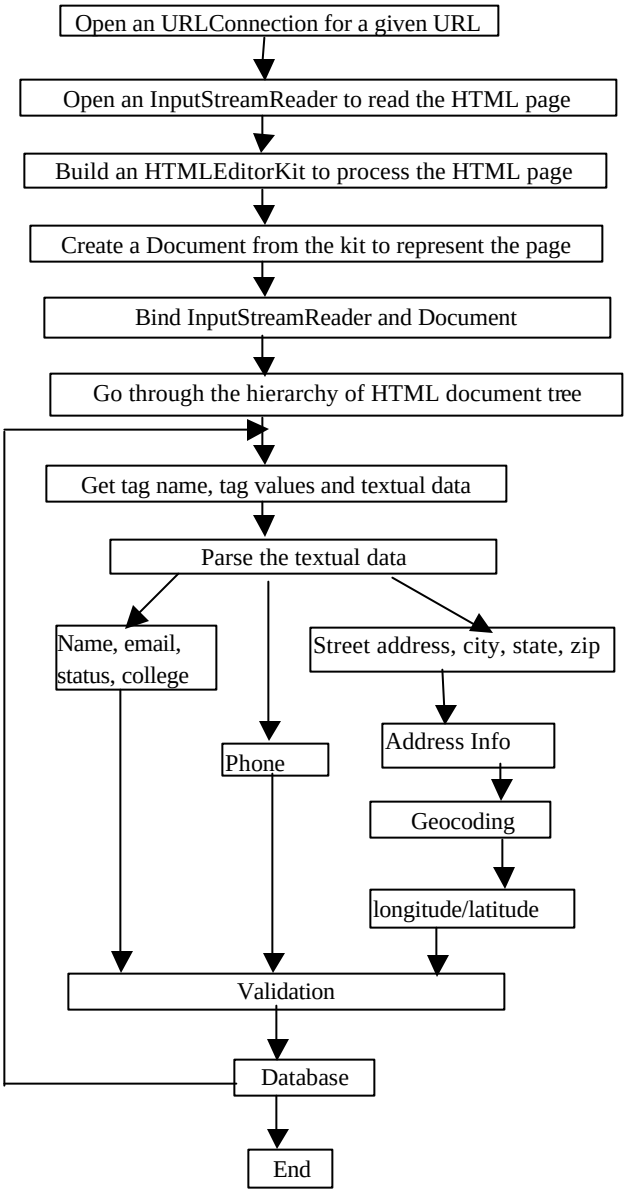


Figure 2. Process of Retrieving Geographical Information from HTML Pages

Generic relational database systems, such as Microsoft SQL Server, do not support user defined data types. For the distributed server that uses SQL Server, the coordination information is stored in two separate table columns. Most Object-Relational databases, such as Oracle 8i ([HREF 19]), either support user defined data types or have predefined object types. In our implementation, for the

distributed server that uses Oracle 8i, the coordination information is stored as the Point that is a predefined object type in Oracle 8i Spatial.

### *GML Wrapper for Relational Databases and Object-Relational Databases*

To build a wrapper that converts geographical data from databases to GML for both relational database and object-oriented databases, a Java Servlet similar to our previous work in wrapping flat files to GeoXML is developed ([10]). The wrapper Servlet first communicates with SQL Server /Oracle 8i through JDBC-ODBC bridge or Oracle JDBC to retrieve the query results and then transfers the query results into a GML document.

Since SQL Server does not have any extensions to support spatial data types and does not support GML directly either, we have to retrieve the student information from the SQL Server database record-by-record and add it to the GML document tree. Also, since SQL server does not support spatial queries, the mediator must communicate with it constantly to retrieve the newly added student address information and store it in the mediator's database which supports spatial queries. The mediator performs spatial queries on the wrapper's behalf in this case.

Oracle is a well-known object-relational database. Although Oracle spatial option stores geographical data in its relational tables, it allows the user to store and retrieve geographical objects such as point, polyline and polygon. It provides spatial access methods such as R-tree, and thus is able to perform spatial query and analysis. The client and the mediator can issue any SQL query to Oracle and get results back in GML format.

The advantages for wrappers to support spatial queries are: (1) there is no duplication of data between the mediator and the distributed server since no mirror database is needed. (2) The client or the mediator always gets the most recent data. (3) Data transfer overhead is also reduced since the resulted GML documents of specific queries are much smaller than retrieving all the data from a wrapper at once.

### *Building a Spatial-Enabled Mediator*

We build the mediator on top of Oracle 8i with Spatial Option. The mediator stores the metadata of all the distributed servers, such as the URLs of the distributed servers, categories or regions of the Web sites that they are responsible to roam over, whether they are active or not, whether they support spatial queries, and when the last time they communicate with the mediator. The mediator also has tables to store geographical entities and their associated student information from those distributed servers that do not support spatial query and analysis.

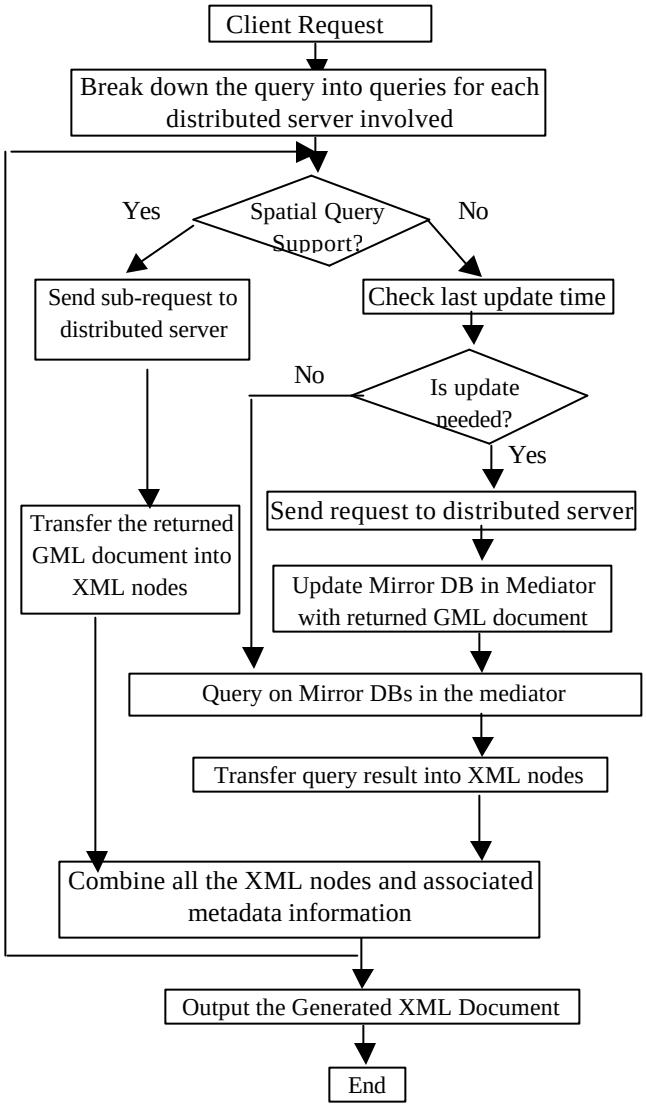


Figure 3. Process of building GML Mediator

The process of builing the GML mediator is shown in Fig. 3. First the mediator accepts a client request and breaks it down into sub-queries to the related distributed servers according to the metadata stored in the mediator. For each sub-query, the mediator checks whether the corresponding distributed server supports spatial queries. If it does support spatial queries, then the mediator simply sends the sub-query to the distributed server and gets the returned GML document. If not, the mediator

checks the most recently data received from that distributed server in its metadata table. If the time span is within a predefined tolerance, the mediator queries the duplicated data on that distributed server's behalf in its own database; otherwise it retrieves the updated data from the distributed server and stores it in a mirror database of the database in the corresponding distributed server. The mediator then combines the query results and sends the integrated GRI also in the GML format back to the requested client.

### *Client Query and visualization*

We build a client query and visualization Applet on top of GeoTools ([HREF 23]), an open source software developed at Center for Computational Geography, University of Leeds, UK. The Applet first sends a request to the mediator using the standard HTTP protocol taking the query string as a parameter in the URL. The Applet then parses the returned GML document. It sends geometric data to the GeoTools and sends textual attribute data to a list box for display. The Applet maintains the links between geometric data and attribute data. This will enable the user to query spatial locations based on attributes, such as "Show where Jacob A Baccus is located", and query attributes based on spatial relationship, such as "List the phone numbers of freshman students whose location is within 5 miles of the campus."

Currently, the client side Applet supports Zoom In/Zoom Out and Full extent of geometric data for visualization and attribute listing of textual data. Also query results can be overlapped on base maps (mostly administrative boundary maps and/or street maps at different scales) that can come from either another GML document or directly from flat geographical data files or spatial databases. In our current implementation, whenever the client issues a new query, the data must be reloaded from the mediator according to the new query. Note that this is not necessary in some cases, such as when the data involved in the new query is only a subset of the result of a previous query since the client can perform queries on the client side. We will implement this feature in our future work.

## Experiments and Results

We test our prototype system on Windows NT 4.0 platform using JavaServer WDK (Web Development Kit) 1.0 as the Web server and Java SDK Standard Edition Version 1.3 as the compiler and runtime environment. We use different virtual directories in the same machine to simulate distributed servers and the mediator. We test only a subset of student contact information at the University of Oklahoma. The agent of the first distributed server retrieves the contact information of 198 students whose last name begin with a letter "A" and stores it in the local database associated with the first distributed server. Similarly, the agent of the second distributed server retrieves the contact information of students whose last name begin with a letter "B" and stores the data in the local database of the second distributed server; there are 519 such students. From the 198 students in the first distributed server and the 519 students in the second distributed server, we eliminate the data of those students who do not have a contact address or their addresses can not be geocoded, i.e., their textual address information can not be converted into coordination. Some possible reasons for this to occur could be that the city name cannot be correctly identified or there is a mismatch between zip code and city/street.

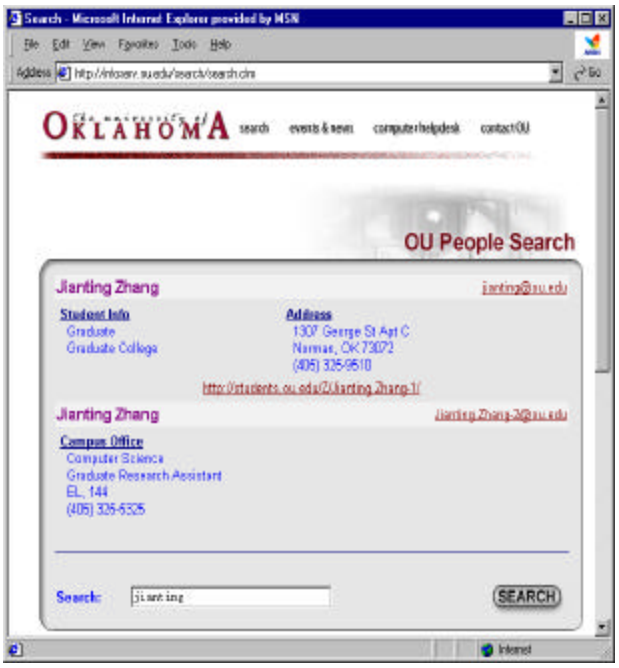


Figure 4. Document Structure of Student Information Query Result

Fig. 4 shows the document structure of the student information query results returned from the OU text-based search engine. Figure 5 shows the wrapped GML document containing the contact information retrieved from the second distributed

server for all students whose last name begin with a letter “B” and displayed in the IE5.5 browser.

Figure 5 Wrapped GML document for Geocoded Student Information

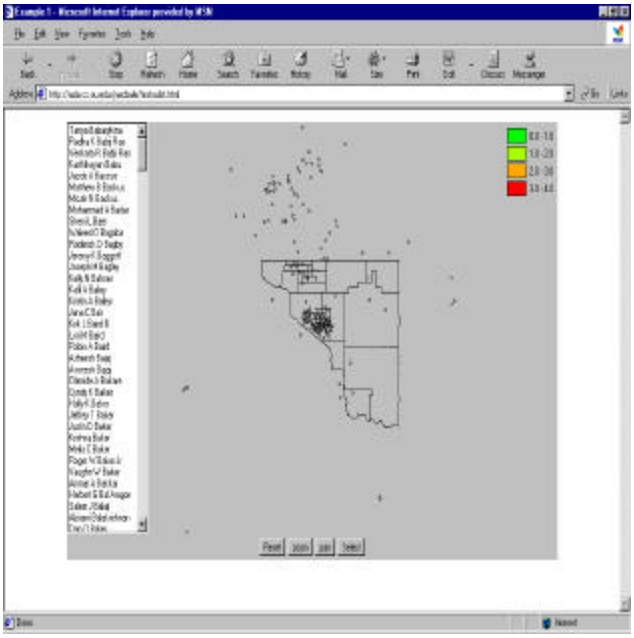

Figure 6 Client Visualization Interface

Due to space limitation, we only show one snapshot of client visualization interface (Fig. 6). We can see that the majority of the students live very near to the campus. However, it is also easy to draw the conclusion that a considerable amount of students put their on-campus dormitories as their contact address which are largely overlapped. Developing more sophisticated tools for client side visualization and query interface is under consideration.

## Conclusions and Future Work

The volume of personal geographical information will be dramatically increased in the near future due to the wide application of GPS devices and cell phones. It is highly likely that this information will be used beyond the centralized systems where it is generated. It will be integrated into the mainstream Web HTML files in a loosely coupled and distributed fashion. Thus a geographical information search engine over the Internet similar to popular text-based information search engines is highly desirable.

In this paper, we have done some preliminary work on proposing a GML-based open architecture and a simple prototype implementation. Specifically, we have done the following:

1) We proposed an open architecture to collect geographical information using the software agent technology and GML as the common communication protocol. We also proposed a negotiation mechanism between the mediator and distributed servers.
2) We built GML wrappers for both pure relational database systems like SQL Server 2000 and object-relational database systems like Oracle 8.1.7.
3) We built a mediator server with full spatial query capability by making full use of Oracle Spatial functionalities.
4) We built client visualization interface for query results using the open source package GeoTools.

Future work directions may include the following:

1) Parsing more generic HTML pages and searching for geographical information in them. Semi-structured document processing techniques (including Information Extraction and Information Retrieval) as well as natural language processing technologies (NLP) are needed for accurate and efficient geographical information searching.
2) Geocoding methods, which are very important for geographical information transformation and

query yet leave untouched in this paper, need to be fully implemented in the system. It is even more challenging when multiple languages are considered since the expression of addresses is highly related to language and culture in general.

3) Spatial and temporal representation of geographical information. Currently no commercial database system support spatial and temporal information of an object. We need to investigate more on Oracle as well as other database systems to find a way to represent both spatial and temporal aspects of geographical information to support advanced queries. A natural extension of this research direction is how to deal with semantics of geographical information that has become increasingly important.
4) Scalability study. Considering the huge amount of Web pages, further investigations on the scalability of the database systems is an important issue.
5) More flexible and better client query and visualization functions.

## References


[1] C. Baru, A. Gupta, B. Ludaescher, R. Marciano, Y. Papakonstantinou, P. Velikhov, "XML-Based Information Mediation with MIX", ACM SIGMOD 99, Philadelphia, PA, USA, 597-599

[2] W.J. Drummond, "Address Matching: GIS Technology for Mapping Human Activity Patterns." J. of the American Planning Association. Spring 1995, 61(3), 240-251

[3] A. Hardie, 1998, "The Development and Present State of Web-GIS, Cartography", 27(2), 1998, 11-26

[4] A. C. Ikeji, F. Fotouhi, "An Adaptive Real-Time Web Search Engine", WIDM 99, Kansas City, Mo, USA, 12-16

[5] W. Lam, M. Ruiz, P. Srinivasan, "Automatic Text Categorization and Its Application to Text Retrieval", IEEE Transactions on Knowledge and Data Engineering, 11(6), 1999,865-879

[6] A. Kruger, C. L. Giles, F. M. Coetzee, E. Glover, G. W. Flake, S. Lawrence, C. Omlin, "DEADLINER: Building a New Niche Search Engine", CIKM 2000, McLean, VA, USA, 272-281

[7] S. Shekhar, S. Ravada, X Liu, "'Spatial-Databases-Accomplishments and Research Needs", IEEE transaction on Knowledge and Data Engineering, 11(1), 1999, 45-55

[8] C. Silverstein, J.O. Pedersen, "Almost-constant-time clustering of arbitrary corpus subsets", ACMSIGIR 1997, Philadelphia, PA, USA, 60-66

[9] I. Zaslavsky, R. Marciano, A. Gupta, C. Baru, "XML-based Spatial Data Mediation Infrastructure for Global Interoperability", in 4th Global Spatial Data Infrastructure Conference

[10] J. Zhang, M.S. Javed, A. Shaheen, Le Gruenwald, "A Prototype for Wrapping and Visualizing Geo-Referenced Data in Distributed Environments Using the XML Technology", ACMGIS 2000, McLean, VA, USA, 27-32

[HREF 1] The Proposed .geo Top-Level Domain Name executive summary, http://www.dotgeo.org/summary.html
[HREF 2] Google, http://www.google.com
[HREF 3] Altavista, http://www.altavista.com/
[HREF 4] MapQuest, http://www.mapquest.com
[HREF 5] Excite, http://maps.excite.com
[HREF 6] ESRI, http://www.esri.com
[HREF 7] MapInfo, http://www.mapinfo.com
[HREF 8] InterGraph, http://www.intergraph.com
[HREF 9]AutoDesk, http://www.autodesk.com
[HREF 10] Alexandria Digital Library Project, http://www.alexandria.ucsb.edu/
[HREF 11] OpenWave, http://www.openwave.com
[HREF 12] The Internet Corporation for Assigned Names and Numbers (ICANN), http://www.icann.org/
[HREF 13] DotGeo, http://www.dotgeo.org/
[HREF 14] Google Map Finder: http://www.google.com/help/features.html
[HREF 15] Centrus, http://www.centrus.com/
[HREF 16] SearchSoftware USA, http://www.searchsoftware.com,
[HREF 17] Open GIS Consortium, http://www.opengis.org/
[HREF 18] Open Geocoding Specification, http://www.ionicsoft.com/geocoding/geoc_schema.html
[HREF 19] Oracle Spatial User's Guide and Reference http://technet.oracle.com/docs/products/oracle8i/doc_library/817_doc/inter.817/a85337/toc.htm
[HREF 20] Geography Markup Language (GML) v2.0, http://www.opengis.net/gml/01-029/GML2.html
[HREF 21] OU Student Information Search Engine: http://infoserv.ou.edu/search/
[HREF 22] Java SDK Standard Edition Version 1.3, http://java.sun.com/j2se/1.3/
[HREF 23] GeoTool Open Source Software, http://sourceforge.net/projects/geotools/