JKAU: Eng. Sci., **Vol. 23** No. 1, pp: 39-57 (2012 A.D. /1433 A.H.)

DOI: 10.4197 / Eng. 23-1.3

Geographic Information System Based on Mobile Agent and Ontology

Ashwaq Maghraby, Mostafa Saleh and **Fathy Eassa**

Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia

ashwaqm@gmail.com, msherbini@kau.edu.sa, Fathy55@yahoo.com

Abstract. A Geographic Information System (GIS) is an important distributed information resource. GIS technology is being utilized in many areas of research. Based on the concept of agent, we introduce a multi-agent architecture to develop a GIS Multi-Agent System based on Ontology (GISMAO) that enables cooperation to assist different users to locate and retrieve spatial GIS file information in large networks based on ontological relationship between GIS concepts. GISMAO consists of two subsystems: Gathering subsystem; and Query subsystem. The Gathering Sub-System is used to collect spatial GIS files information from remote locations, and to store them in the server. Ontology is used to represent concepts and relations that are common concepts in the domain. The Query Sub-System is used to help the user in his/her daily work with the GIS; it works as a GIS retrieval (query) tool. The objective of this system is to demonstrate how the agents can cooperate to transparently locate and retrieve GIS information. To prove the concept, a partial implementation of this framework has been carried out using some modules and libraries of Java, Aglet, Jena and AltovaXML.

Keywords: Geographical Information System, GIS Ontology, Multi-Agent System, Semantic query, Mobile Agent.

1. Introduction

A GIS is a computer system for capturing, storing, checking, integrating, manipulating, analyzing and displaying data related to positions on the Earth's surface. Typically, a Geographical Information System (or Spatial Information System) is used to handle maps of one kind or another. These might be represented as several different layers where each layer holds data about a particular kind of feature. Each feature is linked to a position on the graphical image of a map such as Electrical map layer, Gas station map layer[1-6].

There are two formats used by GIS systems to represent geographical information in digital form: Vector and Raster. Raster data is an abstraction of the real world where spatial data is expressed as a matrix of cells or pixels. The most popular format of a raster file is an image file. Vector data is comprised of lines or arcs, defined by beginning and end points, which meet at nodes. Points are stored using the coordinates system. The most popular formats of vector data are Shapefile, MapInfo and AutoCAD [7-9].

Currently, the GIS technology is being used around the world for daily tasks by many specialists, such as meteorologists, geologists, hydrologists, economists, demographers, disaster relief workers, wildlife conservationists, ecologists, foresters, botanists, and plant breeders, in addition to cartographers and geographers. GIS tools allow these users to rapidly organize, integrate, and visualize various kinds of biophysical and socioeconomic data. The resulting electronic maps serve as a workspace for creative analysis and problem solving [10, 11].

Several problems have arisen in the areas of searching, locating, manipulating and retrieving geographical information where the creation and use of Mobile agents may be successful:

1. **Location and retrieval of Spatial Information in large networks**. To locate and retrieve the right geographic information from the Network we have to create structures that help us in locating and retrieving that information. We can identify the need for creating different agents to locate the necessary information for the user and filter the retrieved information by identifying the user's necessities [12-14].

2. **Facilitating the handling of a GIS user interface**. An Interface agent helps the user in his/her daily work with the GIS, it listens to the user's commands and builds a profile based on what it has gathered, the agent can also build a knowledge base of the way in which he/she works, the agent can also modify the user environment according to his/her preferred tools or the commands that are executed most often, the agent should be able to execute tasks on the user's behalf or suggest actions for them to take [12].

3. **Implementation of improved spatial tasks**. Agents can help in improved spatial tasks through the generation of templates for executed tasks, monitoring of these tasks and creation of intelligent spatial data that let new information become available (locally/ globally)[12].

4. **Creating interfaces between GIS and specific software packages**. Agents can provide an input interface for external commands. This means they will be responsible for communicating with the external package(spatial models, statistical packages, generalization algorithms or any other application) and will finally integrate the results into the GIS data model, they can also add power to the packages through personalization [12].

The goal of this paper is to solve some of GIS problems through the creation and use of multiple agents and ontology. Specifically, this research will focus on two of the above mentioned problems by finding a solution for the location and retrieval of spatial information in large networks and creating a GIS user interface.

A Software Agent is a computer program which works toward goals in a dynamic environment on behalf of another entity (human or computational), possibly over an extended period of time, without the direct intervention of humans or other agents, and exhibits some level of the key attributes of learning, co-operation and mobility [15, 16].

Ontology refers to how people commonly think about a real thing in the world. It clarifies the structure of knowledge, and leads to coherent knowledge base. It is used in artificial intelligence, semantic web, and software engineering as a form of knowledge representation about the world or some part of it [14,17-20]. Ontologies may enable knowledge sharing and reuse for different domains. Geospatial information retrieval and filtering, intelligent geospatial search engines, knowledge discovery, decision model assessment and optimization are typical applications based on using agents or ontology in GIS.

Viegase and Soares [21] presented an ontology-based mechanism that allows different community's users access the same geographic database without knowing its internal structure. They developed ontologies for the coral reef domain which can be used as a navigation and query tool for the users, supplying the semantics information desired.

Nolan, *et al.*[22] developed an ontology and communication language(ACL) for an agent-based GIS by focusing on the three critical components necessary to exchange information between GIS-domain agents, systems or organizations, including: data (vector, raster and image); algorithms descriptions(name, inputs, outputs, and required parameters); and user-posed query/result information. The ontology and ACL let clients and agents search for other agents using implementation independent semantics. They represented entomology and ACL in the Resource Description Framework (RDF) encoded in the eXtensible Markup Language (XML).

Yergens and *et al.* [23] have demonstrated an approach for constructing agent-based simulation models from existing geographical information systems data known as IDESS (Infectious Disease Epidemic Simulation System). This approach is flexible in its ability to model any geographical location for an emerging infectious disease outbreak. This flexibility addresses the unpredictable nature of the geospatial location of where an infectious disease outbreak will occur. The IDESS has shown its ability to rapidly construct an agent based simulation model that can be used in the investigation of the spread of diseases in a given geographical environment.

The authors in Ref. [24-26] presented GeoAgent, which is a mobile agents system working in the geospatial information field in distributed networking environments (internet). This agent acts on the environments with different appearances, to realize intelligent acquirements, processing, storage, exploration, presentation and decision-making support for geospatial data. D. Li *et al.* [24] presented an example of Geo-Agents (overlay operation). An overlay operation overlays two or more map-layers following some particular rules. This paper shows how Geo-Agents improve the performance of the system by reducing data transmission and making use of all of the host's resource available as sufficiently as possible (maximum use of all hosts in the system).

This paper presents a multi-agent architecture, GIS Multi-Agent System based on Ontology (GISMAO), in which agents communicate with each other in order to locate and retrieve GIS vector/raster files information from public and private networks. The agents use GIS ontology to intelligently seek, retrieve, and integrate information to answer complex queries. It is very important that the users be able to obtain timely and complete information without having to know the details of where the relevant data resides and how it is accessed.

This paper is organized as follows: Section 2 presents the GISMAO architecture. Section 3 is dedicated to discuss the GIS Ontology. Section 4 discusses the Semantic Query (Concept-based) Example. Finally, section 5 presents the conclusion and the future work.

2. GISMAO Architecture

GIS Multi-Agent System based on Ontology (GISMAO) is a Multi-agent system, which helps the user to find specific GIS file by using ontology. It is composed of a group of software agents, which are capable of collaboratively completing a task in the GIS domain. These agents run on the same physical machine or they can be distributed across a network (working as distributed systems). As shown in Fig. 1, the GISMAO system consists of two sub-systems: Gathering Sub-System and Query Sub-System. The Gathering Sub-System collects information about GIS data from remote sites in a distributed network and then stores the collected information in the database. It stores concepts and relationships about the GIS information in the ontology of the sub-system. The Query Sub-System helps the users to retrieve information about GIS files directly from the database of Gathering Sub-System.

2.1 *Gathering Sub-System of GISMAO System*

The Gathering Sub-System is a spatial GIS information server, which collects information in such a way that agents will be able to find it and present its value to their users. A database is used to store the information gathered by the Gathering Sub-System agents. Users can search based on semantic, as the concepts and relationships are created and stored in the Main Ontology.

There are two categories of agents: Static and Mobile category. The static category includes *Spatial Task Agents*. The *Spatial task agent*s sub-category consists of four agents: *Sleep Agent*, *Creator Agent, Gathering Save Result Agent*, and *Static ontology Agent*, each one is responsible for doing different job. The *Sleep Agent* is responsible for sending messages to the *Creator Agent* to wake it up every N days. The *Creator Agent* is responsible for creating the *Mobile Agents*. The *Gathering Save Result Agent* is responsible for saving the result (GIS files information) in the *Main DB* (inserting or updating). The *Static ontology Agent* is responsible for handling semantic based query by using the Main Ontology.

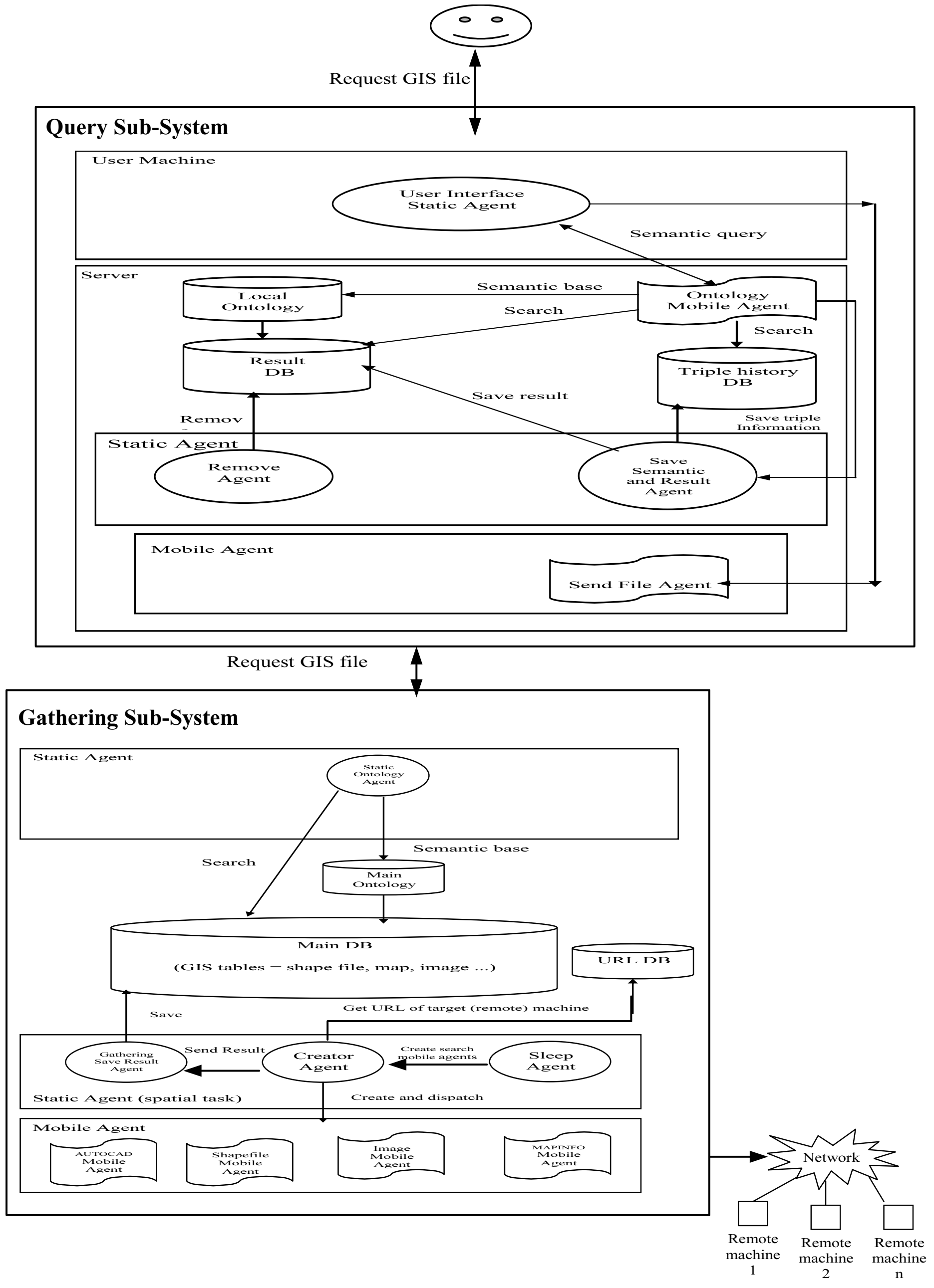

Fig. 1. GISMAO general architecture.

The Mobile category consists of four mobile agents: *Shapefile Mobile Agent, Image Mobile Agent, AUTOCAD Mobile Agent,* and *MapInfo Mobile Agent*. Each one of these agent travels through the network every **N** days (such as every 7 days), collects information about vector/raster GIS files and saves the information in the Main DB.

2.2 *Query Sub-System*

The Query Sub-System is used to help the user in his/her daily work with the GIS. It works as a GIS query tool. It is composed of a group of software agents, which run on the same physical machine or distributed across a computer network (homogeneous or heterogeneous).

There are two categories of agents: Static and Mobile. The Static category includes three agents: *User Interface Static Agent*, *Save Semantic & Result Agent* and *Remove Agent*; each one is responsible for doing different tasks. Each agent is responsible for working with one or more of the following: the user, Result DB, ontology and other agents. The GISMAO communicates with the human users *via* the *User Interface Static Agent* (*UI Agent*). If the user wants the help of the GISMAO, he/she writes GIS files question (semantic query), then the user will pass it on to the *UI Agent*. The *UI Agent* will do everything while the user simply waits for the result (s).

The *Save Semantic & Result Agent* and the *Remove Agent* are working with the Result DB. The *Save Semantic & Result Agent* is responsible for saving GIS files information in the Result DB (inserting or updating) and saving triples information in the Triple History DB. The *Remove Agent* is responsible for removing files from the local machine and keeping its information after M days (such as 30 days).

Two Mobile Agents travel through the network to the remote machine to find GIS information. One of them is the *Ontology Mobile Agent*, which is responsible for handling semantic based queries by using ontology. The *Ontology Mobile Agent* uses the Local Ontology to analyze semantic queries and offers triples. After that, it searches in the Triple History DB to finds the same triples. If it finds the triples, it gets the queries of these triples from the Triple History DB and searches in the Result DB to find file information. If it does not find the triples in the Triple History DB, it travels to the Gathering Sub-System collaborating with the *Static Ontology Agent* to get information about special GIS files.

The other mobile agent is the *Send File Agent*, which travels to the remote machine to get special GIS files.

3. GIS Ontology

The GISMAO provides the means to find particular vector/raster layers or maps, which have special attributes (properties), objective and city. The GISMAO processes data to answer questions such as: "What objective of this map", "What is the objective of this layer", "What is the Coordinate System of PLAY GROUND layer in USA city", "Give me layers of Jeddah city", "Show PARK layer of Jeddah city" and others. These questions are called semantic based queries. The GISMAO handles semantic based queries by using GIS ontology (Main Ontology). Ontology concerns objects, relations, states, events and processes in space [26]. The ontological approach clarifies the structure of knowledge, and leads to consistent knowledge base. It consists of object classes with attributes representing state and with relations between objects and attributes [27, 28].

GIS ontology promises a shared and common understanding of GIS domain that can be communicated between people and application systems. It enables knowledge sharing and reuse for different domains. It covers fields ranging from knowledge engineering, information integration, information retrieval, and object-oriented analysis. The Gathering Sub-System handles semantic based queries by using Static Ontology Agent. Static Ontology Agent analyzes many different data such as Layer Name, Layer type, Map Name, Map type, objective using Main Ontology.

The Main Ontology of the GISMAO determines what is of interest in the GIS domain and how its information is structured. Figure 2 shows the Main Ontology. This ontology consists of 17 basic concepts (Map, Layer, Raster Layer, Vector Layer, Image Layer, Shapefile Layer, MapInfo Layer, AutoCAD Layer, and AutoCAD Layer with Coordinate System, AutoCAD Layer without Coordinate System, Machine, Network, City, Objective, Humana, GIS Application Developer, Map or Layer Developer). This ontology also consists of 8 basic relations (Sub-Class of, Create, Compose of, Describe, Use, Located at, has Objective).

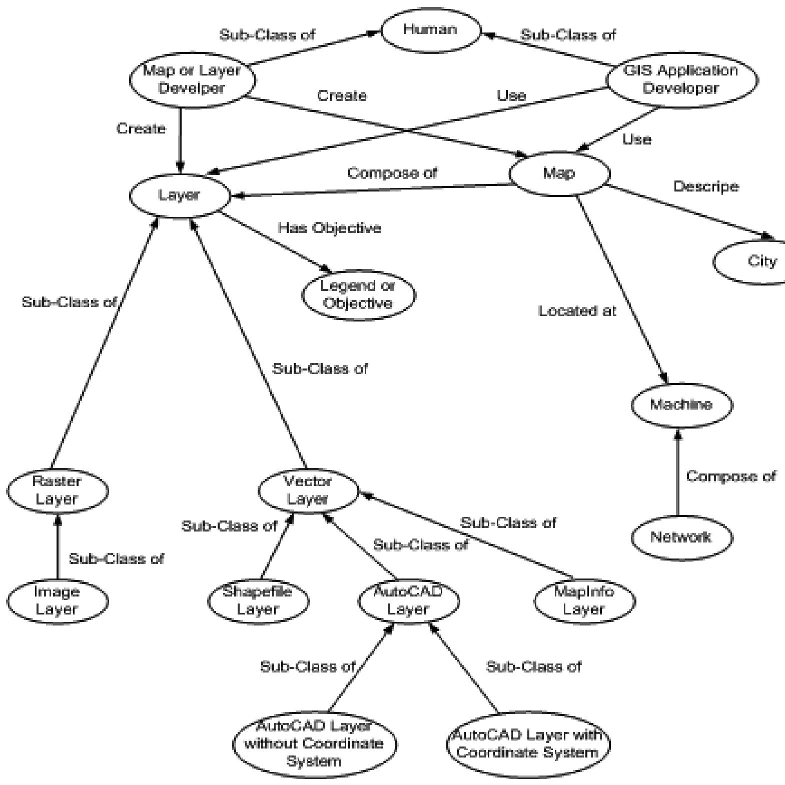

Fig. 2. GIS ontology (main ontology).

The Query Sub-System of the GISMAO helps the users to retrieve information about GIS file based on the semantic of query by using ontology. It handles semantic based queries by using the Ontology Mobile Agent, Local Ontology, and Main Ontology.

The Query Sub-System has its own ontology (Local Ontology) that helps it to timely and locally handle the semantic based requests that are similar to previously requested queries. Therefore, the Query Sub-System avoids the cost of the communication between agents and the cost of data transmission. It also avoids the huge spaces that is held by the ontology (Main Ontology), since it can hold sub set of the Main Ontology that provides the GIS information that the user is interested in (if user is interested in shape file information only, the Query Sub-System holds shape file ontology only as Local Ontology). The GIS ontology is implemented using RDF/RDFS in Fig. 3.

```
<rdf:RDF
xmlns:rdf="http://www.w3.org/
1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org
/2000/01/rdf-schema#">
<rdfs:Class rdf:ID="Map"/>
<rdfs:Class rdf:ID="Layer"/>
<rdfs:Class rdf:ID="City"/>
<rdfs:Class
rdf:ID="Machine"/>
<rdfs:Class rdf:ID="City"/>
<rdfs:Class
rdf:ID="Network"/>
<rdfs:Class
rdf:ID="RasterLayer">
<rdfs:subClassOf
rdf:resource="#Layer"/>
</rdfs:Class>
<rdfs:Class
rdf:ID="VectorLayer">
<rdfs:subClassOf
rdf:resource="#Layer"/>
</rdfs:Class>
<rdfs:Class
rdf:ID="MapInfoLayer">
<rdfs:subClassOf
rdf:resource="#VectorLayer"/>
</rdfs:Class>
<rdfs:Class
rdf:ID="AutoCADLayer">
<rdfs:subClassOf
rdf:resource="#VectorLayer"/>
</rdfs:Class>
<rdfs:Class
rdf:ID="ShapeFileLayer">
<rdfs:subClassOf
rdf:resource="#VectorLayer"/>
</rdfs:Class>
<rdfs:Class rdf:ID="AutoCAD-
CoordSystem">
<rdfs:subClassOf
rdf:resource="#AutoCADLayer"/>
</rdfs:Class>
<rdfs:Class rdf:ID="AutoCAD-
NoCoordSystem">
<rdfs:subClassOf
rdf:resource="#AutoCADLayer"/>
</rdfs:Class>
<rdf:Property
rdf:ID="ComposeOf">
<rdfs:domain
rdf:resource="#Map"/>
<rdfs:range
rdf:resource="#Layer"/>
</rdf:Property>
<rdf:Property rdf:ID="Describe">
<rdfs:domain
rdf:resource="#Map"/>
<rdfs:range
rdf:resource="#City"/>
</rdf:Property>
<rdf:Property
rdf:ID="LocatedAt">
<rdfs:domain
rdf:resource="#Map"/>
<rdfs:range
rdf:resource="#Machine"/>
</rdf:Property>
<rdf:Property
rdf:ID="ComposedOf">
<rdfs:domain
rdf:resource="#Network"/>
<rdfs:range
rdf:resource="#Machine"/>
</rdf:Property>
</rdf:RDF>
```

Fig. 3. GIS ontology using RDF/RDFS.

To use ontology in GIS systems, we need to migrate the huge legacy systems data into ontological representations and storing this data in ontology based format such as Resource Description Framework

(RDF), and RDFS [29, 30] format. But this technique will duplicate the data in these systems. So, it is better to wrap the available legacy GIS data with ontology and map each concept in ontology with its related database table. This will enable users with two types of queries: tradition query based on the modeling database structure; and semantic query based on the overlay ontological concepts representation.

To enable the previous notion of wrapping traditional GIS data with ontology we need to link each concept in the ontology with its related data table as shown in Fig. 4. This will enable users to query based on their logical representation of concepts neither the SQL query based-on the internal database structure. Consequently, we can find different GIS database structures but comply with same ontology representation.

4. System Implementation Environment

A prototype has been implemented using Java. Sun Microsystems provide Java Development Kits (JSDK) for many platforms, a standard edition and enterprise edition [31]. Concerning RDF, there are many RDF libraries now, but the most complete library is probably Jena from HP Labs [32]. Jena was developed to provide an API that was designed specifically for the Java programming. Jena provides support for serialization, relational database storage and querying. Also, it provides an ontology API and comes with rule engines for basic inference on RDF Schemas. It also has limited support for DAML+OIL and OWL. We used Jena to provide the RDF data sources and querying. Additionally, RDQL, RDF Data Query Language, first released in Jena but currently SPARQL is supported in Jena *via* a module called ARQ. It is a powerful language and simpler than the earlier RDF query languages which make it the best choice.

AltovaXML is a free XML application package that can be used to validate XML documents, transform XML documents using XSLT style sheets, and execute XQuery documents. Also, it can be used from the command line *via* a COM interface in Java programs and in .NET applications. The Altova XQuery 1.0 Engine conforms to the W3C. This Engine Executes an XQuery document using well-formed XML documents. However, they do not need to be valid according to an XML Schema since the invalid file is loaded without schema information. If the XML file is associated with an external schema and is valid according to it, then post-schema validation information is generated for the XML data and will be used for query evaluation [33].

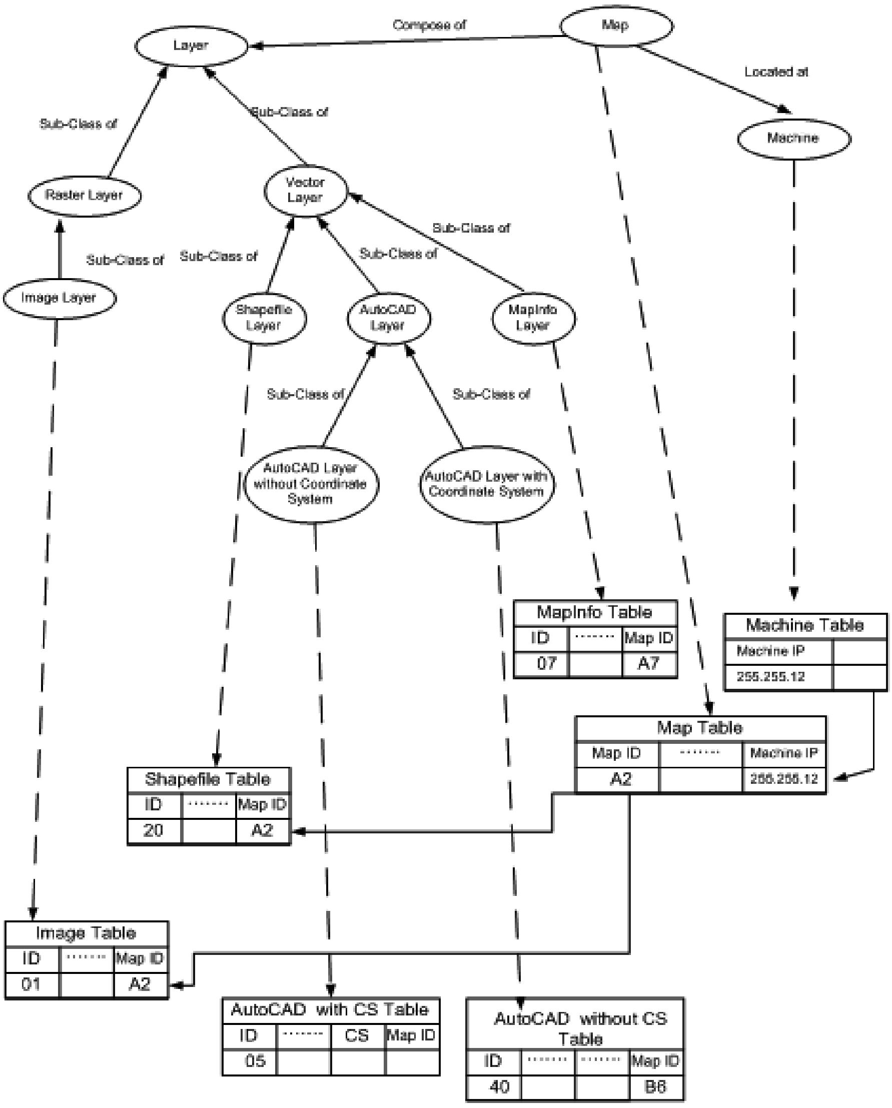

Fig. 4. A Portion of the GIS ontology linked to the database.

Numerous agent platforms have been implemented or are currently under development. One of them that implement many important features of agent is Aglet. Aglet originally developed at IBM Tokyo Research Laboratory and currently the project is hosted at source forge [34]. Aglets are a mobile-agent technology built on top of Java. They are similar to applets in that they run as threads inside the context of a host Java application. Aglets need a host Java application to be running on or they want to immigrate. This host application is called an “aglet host”. The aglet host installs a security manager to implement restrictions on any distrusted aglets. Several versions of Aglet have been released and we obtained the latest version which is Aglet2.0.2 from the web site [35].

5. Semantic Query Example

This section explains an example to query based on semantic. If the user wants to search by: "Jeddah Park Layer", the GISMAO system will go with the following procedure:

At the beginning, the user enters the above statement and clicks on the search button, which passes this statement to the *UI agent* with a *[Start Search]* message.

The *UI Agent* sends the [stat_search] message to the *Ontology Mobile Agent*. If it finds the *Ontology Mobile Agent*, it sends the *[Start Search]* message to the *Ontology Mobile Agent*.

The *Ontology Mobile Agent* uses the *Local Ontology* to analyze the semantic based query and produces triples (Table 1 shows Triples Query). After that, it searches in the *Triple History DB* to finds the same triples (Table 2 shows an example of the *Triple History DB*).

Table 1. Triple Query.

Subject	**Relation**	**Object**
? Map	**Describe**	**City**
Map	description	Jeddah
? Layer	**Describe**	**Objective**
Layer	description	Park

Table 2. Triple History DB.

Triple	**Number of queries**	**queries**
Layer + description + EMBASSY	1	Select all from shapefile table Where LayerName = "21EMB001"
Layer + description + AIRPORT	1	Select all from shapefile table Where LayerName = "21CAC001"

In this example, it does not find the triples in the *Triple History DB*, so the *Send_file Agent* migrates to the *Gathering Sub-System.* When it arrives at the *Gathering Sub-System,* it sends the *[Triple]* message to the *Static Ontology Agent* and waits for result.

Once the *[Triple]* message is received by the *Static Ontology Agent,* the *Static Ontology Agent* gets the triples from this message and uses the *Main Ontology* to determine queries. In this example, the *Static Ontology Agent* searches in the *Map City* table in the Main Ontology to get Map Name (Table 3 shows certain example of the *Map City* table). It searches also in the Layer Objective table in the Main Ontology to get Layer Name (Table 4 shows certain example of the *Layer Objective* table). It gets Map Name ="001" and Layer Name = "23PUB001". The *Static Ontology Agent* forms the query as:

(Select all from Shapefile table

Where Map Name = "001" and Layer Name = "23PUB001”).

Table 5 shows an example of Shapefile Table.

Table 3. Map City Table.

Map Name	Relation	City
001	description	Jeddah
002	description	USA
003	description	Washington
004	description	Mexico

Table 4. Layer Objective Table.

Layer Name	Relation	Objective
21EMB001	description	EMBASSY
21EMB002	description	CONSULATE
21EMB003	description	LEGATION
23PLN001	description	LANDMARK
23PLN002	description	HISTORICAL PLACES
23PUB001	description	PARK
23PUB014	description	AIRPORT
21CAC001	description	SHOPPING CENTER

Table 5. Shapefile Table.

Shapefile ID	Map ID	Map Name	Name	File Size	version	Type	machine	…………..
4180	296	WASHINGTON	23TP001	653 KB		area	atp://saad:9000	
4181	297	USA	22ST001	246 KB		Line	atp://saad:9000	
4182	297	USA	21ROA001	23 KB		point	atp://saad:9000	
4183	297	USA	23AHP001	1 KB		area	atp://saad:9000	
4184	297	USA	21CAC001	217 KB		point	atp://saad:9000	
4185	297	JEDDAH	22ST012	1482 KB		line	atp://saad:9000	

The *Static Ontology Agent* searches in the Main DB using the above query, saves the layer information (*Result array*) and triples information in the *[Result]* message, and sends the *[Result]* message to the *Ontology Mobile Agent*. Table 6 shows the Result array.

Table 6. Result Array.

Index Name	Array Data
Map Name	JEDDAH
Name	22ST012
File Size	1482 KB
Type	line
Number of Attribute	12
Attribute Name	Name, location, x, y …
Number of Records	6524
Coordinate System	GCS_North_American_1927
modified	12/18/97
Remote location	D:\GIS\CODE\MAINSERVERAGENT\SHAPEFILEDATA\USA\22ST002

The *[Result]* message is received by the *Ontology Mobile Agent*. The *Ontology Mobile Agent* travels back to the Query Sub-System machine.

When the *Ontology Mobile Agent* arrives at the *Query Sub-System* machine, it creates the *Save Semantic & Result Agent* and sends the *[Result]* message to the *Save Semantic & Result Agent*. It also sends the *[Result]* message to the *UI Agent* that shows the result to the user in the result window.

The *[Result]* message is received by the *Save Semantic & Result agent*. The *Save Semantic & Result agent* holds the argument objects of this message which mean gets the *Result array*, the triples information and others. Since the search type value is 'shapefile', the *Save Semantic &Result agent* calls some function to search in the shapefile result table in the *Result DB* to find the same result. In this example, it does not find any same result, so it saves the *Result array* in the shapefile result table. After that, the *Save Semantic & Result agent* saves the triples information in the *Triple History DB*.

6. Conclusion and Future Work

This research, presented architecture and prototype for geographic information locating and retrieving based on agent technology and ontology (GIS Multi-Agent System based on Ontology GISMAO). Since GISMAO is a distributed system, it has the following features and advantages: GISMAO is flexible distributed architecture, it reduces communication cost, limits local resource, self starts locating and retrieving GIS information and reduces the locating and retrieving time for GIS users. Also, since GISMAO is a multi-agent system, then all of its components have been implemented as agents (mobile or stationary). Mobile and stationary agents have been built under the same agent server (middleware) to interact together for satisfying the system requirements.

We have identified several avenues for future research. First, we would like to update the the GISMAO to locate and retrieve all GIS vector/raster information of different GIS layers. Finally, we want to implement Main Ontology for GISMAO that will add intelligence to the GISMAO and help users to build complex queries during the interaction.

References

[1] **Goodchild, M.F., Maguire, D.J., Rhind, D.W.** and **Longley, P.A.**, *Geographic Information Systems and Science*, 2nd Edition, Chapter 8 (2005).

[2] **Chang, K.S.,** *Introduction to Geographic Information Systems*, 3rd Edition, Chapters 3-5 (2006).

[3] **Pidwirny, M.,** *Introduction to Geographic Information Systems,* Fundamentals of Physical Geography, 2nd Edition, University of British Columbia Okanagan (2006). http://www.physicalgeography.net/fundamentals/2f.html

[4] **Fei, L.L.** and **He, J.,** Harmonic generalization based on the integrated geographic feature retrieval*, International Symposium on Spatial Analysis, Spatial-Temporal Data Modeling, and Data Mining* (2009).

[5] **Edler, D.,** *Geographic Feature Code Catalogue,* BCGOV ILMB GeoBC Information Services Branch, 2008.

[6] **Ponce, F.J., Pallares, F.M., Juan-Llacer, L.** and **Cardona, N.,** Educational software tool based on a geographical information system (GIS) for radio wave propagation analysis, *Education IEEE Transactions*, **44**(4): 355-364, Nov. (2001).

[7] **Raster, Y. Wu.,** *Vector, and Automated Raster-to-Vector Conversion, in Moving Theory into Practice: Digital Imaging for Libraries and Archives,* Research Libraries Group (2005).

[8] **Reed, S.**, *Digital Imaging Part 1 - The Differences in Raster and Vector Images,* Tucows Inc., 25 Apr. (2006).

[9] **Wu, Y.,** *Raster, Vector and Automated Map Digitizing*, Software Corp. (1999). http://www.ablesw.com

[10] **Wang, J.A.,** *Collaborative GIS Decision Support Model on Internet,* Lockheed-Martin Service Group, Visualization Center of National Environmental Supercomputing Center.

[11] **GIS**: *A Window on Tropical Agriculture and Natural Resources*, Centro Internacional de Agricultura Tropical (2001).

[12] **Rodrigues., A.** and **Raper, J.,** *Defining Spatial Agents,* Spatial Multimedia and Virtual Reality, Research Monographs Series, published by Taylor and Francis, pp: 111-129 (1999). *http://www.dh.lnec.pt/ghi/asrodrig/*

[13] **Li, D., Lei, Q.,** ***et al.***, *Geo-Agents: Design and Implement,* Artificial Intelligence Lab in Department of Computer Science and Technology, Peking University (2003).

[14] [14] **Volz, R., Oberle, D.** and **Studer, R.,** Views for light-weight web ontologies*, Proceedings of ACM Symposium of Applied Computing (SAC)*, pp:160-70, March (2003).

[15] **Krupansky, J.W.,** *What is a Software Agent,* Advancing the Science of Software Agent Technology (2005).

[16] **Green, S., Hurst, Nangle, L.B.** and **Cunningham, P.,** Software Agents: A review, *The Knowledge Engineering Review,* **11**(3): 27 May (1997).

[17] **Maedche, A.** and **Staab, S.,** Ontology learning for the semantic web*, IEEE Intelligent Systems*, **16**(2): 72-79 (2001).

[18] **Fensel, D., Harmelen, F.V., Horrocks, McGuinness, I.D.** and **Patel-Schneider, P.F.,** Oil: An ontology infrastructure for the semantic web, *IEEE Intelligent Systems,* **16**(2): 38-44 (2001).

[19] **Tovinkere, V.** and **Qian, R.J.,** Detecting Semantic Events in Soccer Games: Toward a Complete Solution, *IEEE Intelligent Systems*, **16**(2): 1040-1043, August 2001.

[20] **Migliorati, R.P.**, Semantic indexing of multimedia documents, *IEEE Multimedia*, **9**(2): 44-51 (2002).

[21] [21] **Viegase, R.** and **Soares, V.,** *Querying a Geographic Database Using an Ontology-Based Methodology*, Campus University (2006).

[22] **Simon, J.R.** and **Sood, A.,** *Developing an Ontology and ACL in an Agent-Based GIS,* National Imagery and Mapping Agency University Research Initiative Program (2001).

[23] **Yergens, D., Hiner, J., Denzinger, J.** and **Noseworthy, T.,** *Multi Agent Simulation System for Rapidly Developing Infectious Disease Models in Developing Countries*, Centre for Health and Policy Studies and Department of Computer Science, University of Calgary (2006).

[24] **Li, D., Lei, Q. *et al.*,** *Geo-Agents: Design and Implement*, Artificial Intelligence Lab in Department of Computer Science and Technology, Peking University (2003).

[25] **Yingwei, L., Xiaolin, W.** and **Zhuoqun, X.,** *A Dynamic Load Balancing Policy for Agent-Based DGIS*, Asia GIS Conference (2003).

[26] **Yingwei, L., Xiaolin, W.** and **Zhuoqun, X.,** Agent-Based Collaborative And Paralleled Distributed GIS, Geo-Imagery Bridging Continents, *XX th ISPRS Congress, 12-23 July* (2004).

[27] **Chandrasekan, B., Josephson, J.R.** and **Benjamins, V.R.,** *Ontology of Tasks and Methods,* Knowledge Acquisition Workshop, Banff (1998).

[28] **Kuhn, W.,** Ontologies in Support of Activities in Geographical Space, *International Journal of Geographical Information Science*, **15**(7): 613-631 (2001).

[29] **W3C**, *Resource Description Homepage*, *http://www.w3c.org/rdf.* Last visited Aug. (2010).

[30] **Schmelzr, R., Vandersypen, T., Boolmberg, J., *et al.*,** *XML and web Services Unleashed.* SAMS, p. 1005 (2002).

[31] **Sun Microsystems**. http://java.sun.com. Last visited Aug. 2010.

[32] **Jena Semantic Web Framework**. *http://jena.sourceforge.net,* Jena 2.5.1. Last visited Aug. (2010).

[33] **Altova Solution Center**. *http://www.altova.com.* Program AltovaXML2007 version 9.0.2.0, October 2006. Last visited Aug. (2010).

[34] **Luca Ferrari**. "*The Aglets 2.0.2 User's Manual*". October (2004).

[35] Source Forge open source software development web site. *http://sourceforge.net/project/aglet,* Aglets 2.0.2. Last visited Aug. (2010).

نظام معلومات جغرافي مبني على تقنية الوكيل وعلم علاقات المفاهيم

أشواق مغربى، و مصطفى صالح، و فتحى عيسى
كلية الحاسبات وتقنية المعلومات، جامعة الملك عبدالعزيز
جدة، المملكة العربية السعودية

المستخلص. تعتبر نظم المعلومات الجغرافية (GIS) مصادر مهمة للمعلومات الموزعة، حيث أصبحت تستخدم في العديد من مجالات البحث. واعتمادا على مفهوم الوكيل البرمجي وعلم علاقات المفاهيم (Ontology)، يقدم هذا البحث معمارا مبنيا على نظام وكيل برمجي متعدد للمعلومات الجغرافية (GISMAO) يمكّن الشركات من مساعدة المستخدمين المختلفين لتحديد مكان المعلومات الجغرافية الموجودة على الشبكات الكبيرة واسترجاعها اعتمادا على علاقات المفاهيم بين عناصر GIS. وينقسم نظام GISMAO إلى نظامين فرعيين: نظام التجميع (Gathering Subsystem) ونظام الاستعلام (Query Subsystem). يستعمل نظام التجميع لجمع معلومات ملفات النظم الجغرافية من أجهزة الحاسوب البعيدة وتخزينها في جهاز الخادم. يتم استخدام علم علاقات المفاهيم لتمثيل معلومات الملفات الجغرافية وعلاقاتها. أما نظام الاستعلام فيستخدم كأداة لمساعدة المستخدمين في عملهم اليومي مع نظم المعلومات الجغرافية. يهدف النظام إلى توضيح تعاون الوكلاء البرمجية لإيجاد واسترجاع المعلومات الجغرافية. وتم برمجة نموذج مبدئي للنظام بلغة جاف وأدوات Aglet و Jena و AltovaXML.