# Semantic Emergency Management Architecture (SEMA)

Salman Tahir Qureshi*(salmantq@yahoo.com) and Syed Z. Arshad
SZABIST
Karachi, Pakistan

***Abstract**: The World is not safe from disaster's which comes without any warning. Only pre-planning and carefully measured actions at the right time can help disaster affected region. Sufficient resources are required for humanitarian operations in disaster affected region. Effective emergency systems which provide reliable information at right time can save hundreds of lives. In times of emergency, the extraction of relevant emergency related data that is scattered around the globe becomes complicated due to the heterogeneity and scattered nature of data sources. The proposed solution, SEMA is an architecture, which assists in managing emergency relevant information effectively. The architecture allows the unified representation and manipulation of diverse spatial data sources. It provides data integration by enabling actual co-operation among a set of autonomous and heterogeneous information sources in form of Agent. This paper also describes concepts and relationship of emergency management architecture by means of ontology which will be used by agents to understand this particular domain. The agents cooperate and interact amongst themselves and form a hybrid multi-agent environment.*

***Keywords:** Disaster, Emergency management, Web services, Ontologies, Evolution, Multi-agents, GIS, Feasibility, Viability, Semantic Links*

## 1. INTRODUCTION

Natural disasters are a great threat to any country where the infrastructure is weak and one needs to be prepared with modern tools and systems to face disaster. As a result, when these countries face a disaster they incur a huge loss of not only human lives but also of other valuable assets e.g. communication loss, infrastructure damage etc. To support the point we can take the example of Pakistan. It has suffered from numerous natural and human induced disasters and emergencies like floods, earthquakes, landslides, cyclones, drought, fires, civil unrest and terrorism, refugees and internally displaced people, health epidemics, transport accidents, industrial accidents and war. The human impact of natural disasters in Pakistan can be judged by the fact that 6,037 people were killed and 8,989,631 affected in the period 1993 to 2002 [1]. Following the South Asia earthquake, 17,000 disaster-affected women in Pakistan were estimated to be about to give birth. Around 1,200 would face major complications and 400 would require surgery. Yet there was a critical lack of female doctors and health workers [2].

During any kind of emergency in Pakistan, responsible authorities have been unable to properly manage the situation. One of the major reasons behind this performance of organizations is the lack of knowledge management tools. To deal with this problem, Pakistan needs to have tools and applications that must help it in gathering, analyzing and presenting information at the right time. After the October 8, 2005 earthquake, the Government has become concerned and many developments have taken place to improve the Disaster/Emergency Management System of the country. Microsoft is developing a Disaster Data Management System for Pakistan which would have a three phase strategy [3]. Sahana Phase I is being deployed in Pakistan together with the support of NADRA and IBM Pakistan [4]. This leads us to conclude the fact that the aftermaths of these disasters are huge and in order to cope with them we need proper disaster management systems.

## 2. PROBLEM STATEMENT

To cope with disaster situation agencies, government public health sector, private hospitals and international community need up-to-date data regarding current situation in the affected area. We need a system which can accommodate all information regarding a disaster hit region which includes Agency tracking, camp registry and request management system. Agency tracking will monitor all agencies working in the area. Camp registry will store the information related to missing, injured, replaced and displaced people. Request management will help to keep track of supply and demand of aid. The learning algorithm will help in creation of anthologies which will be used by agents to perform above functionalities.

* Quality Assurance Analyst at ARPATECH Pvt Ltd and enrolled in SZABIST MS(SE)

### 3. METHODOLOGY

Our research methodology is speculative, empirical and evolutionary. We have devised an architecture prototype and now this architecture is open for implementation and other future development work. This paper approach is using multi-agents collaboration in a distributed environment. We further found ontology's to be the most proven way of storing data as their evolution is easy and traceable. Multiple approaches of managing spatial data were studied and after due consideration reached a common consensus about combining the multi-agent architecture and ontology evolution framework with the web services.

### 4. SOLUTION: SEMANTIDE

We devised a Semantic Web Emergency Management System [SEMA] as a solution to the existing problem of scattered nature of spatial and heterogeneous data. SEMA is the solution in which data sources and services are made available through web services described by the ontology, communicated through agents allowing interoperability and reasoning to create an ample response adapted to user objectives.

### 5. SEMA HIGH LEVEL ARCHITECTURE

SEMA has been designed in a four layer architecture which includes the following.

#### 5.1. Legacy System Layer

This layer provides data and functionalities of existing legacy systems. It is the basis from where the data actually comes. It can be located on multiple locations across the globe.

- *Meteorological Office* Services*:* Provides weather information (e.g. snowfall, rain) in specific spatial areas.
- *Emergency Planning Services:* Provides information about primary and temporary rest centers, hotels, inns, hospitals, and supermarkets in emergency areas.
- *Buddy Space:* Retrieves the list of presences of appropriate officials and authorities in emergency areas.

#### 5.2. Service Abstraction Layer

This layer depicts the data and functionalities provided by the legacy system layer as Web Services (WS). Web Services act as communication path between the agent architecture and the legacy system data. Web services help in transportation of data from legacy system and then forwarding it to their consequent agents.

#### 5.3. Agent Layer

The agent layer mainly consists of two kinds of agents:

##### 5.3.1. Local Learner Agent & Teacher Agent

Any Local Learner Agent (LLA) can act as a Teacher Agent (TA) provided it holds the specific concept for which it is queried. The LLA holds the functionality for playing the role of a knowledge provider for other agents. LLA's accesses ontology explanation through their inbuilt ontology component. At startup time they load their unique base ontology, and try to use it as long as it caters to fulfill its purpose. The simplest method for an agent to figure out that it does not know the concept of the incoming query is by tracking the incoming queries and matching it with its base ontology component. When a LLA has failed to answer some queries, it tries to find likely consistency among previous unanswered queries. Using this approach and the elements of incoming queries the LLA makes a new query and submits it to Manager Agent (MA) to find out which agents know about the probable concept. The LLA, which acts as a knowledge provider to the other LLA, for the time being acts like a Teacher Agent (TA). Hence, the LLA has two roles. When it requires information it acts as a LLA, but when it acts as a knowledge provider it acts as a TA.

##### 5.3.2. Manager Agent (MA)

The MA has five major roles. First, it acts like a central repository as it holds part of LLA ontology, known as Global Ontology Component (GOC) which is unique for each client LLA. On getting a query the MA first looks for the concepts presence in the GOC. The second function of the MA is to forward the queries to the required LLA which has the answer to the query. So in this case the MA acts as a redirecting agent. For redirecting to the relevant TA the MA looks for the relevant TA which it thinks holds the concept required by going through the GOC of the entire domain specific LLA it holds. One of the limitations of the system is that once the MA finds the required LLA which holds the concept then the actual transfer of data takes place through the MA and cannot be directly transferred without the intervention of the MA. So the third function of the MA is that it provides encoding to the relevant format the other LLA understands. So it provides storing the ontology encoded, translating, interpreting and executing incoming queries, then translating back the results to a format understood by the LLA and forwarding the concepts to the LLA. Translation is necessary because the encoding for storing knowledge and answering queries, which is endorsed by performance requirements might not be the same as the one used in the LLA. The fourth function of the MA is to learn the concept passed by one LLA to the other once it is passed to it for redirection to the other agent. The MA hence updates the GOC of the LLA by adding the newly learned concept to it and at the same time sends a copy of the newly learned concept to the LLA which required the information. The fifth and the last function the MA performs is that once it goes through the GOC of all the LLA in its network and finds no relevant TA then it passes the query to the MA of the other network which might be at some other location on the globe.

### 5.4. Presentation Layer

It is a web interface of the application in which all the goals defined are reflected and once the result of the query reaches the LLA it is displayed on the presentation layer with the help of IRS –III [5].

## 6. WORKING MECHANISM

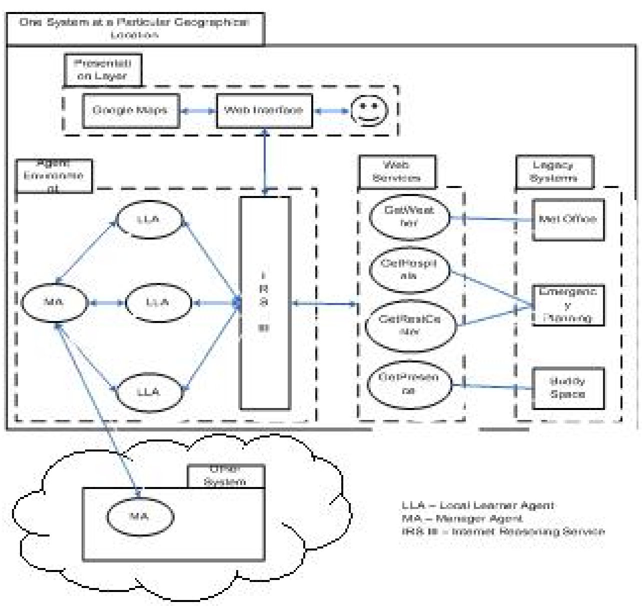


Figure 1- High Level Architecture

Information at the time of emergency needs to be searched analyzed and communicated from relevant sources and locations. An incoming query arrives from the user sitting at one location to the IRS-III. The IRS semantically describes the query which it receives through the web interface. The IRS knows about the agents' domains and hence forwards the query to the best possible LA. The data required by the user might be residing at different locations e.g. met office, emergency planning institutes etc. Once the query arrives from the IRS through a web interface it is passed on to the multi agent system. Each LLA represents one specific information domain like Meteorological Office or Emergency Planning Institute. When the query arrives at a domain specific LLA, the agent looks for the required concepts and answers to the query. Each of the domain specific agents represents all the data of its specific domain prevalent in the Legacy System which is accessible through the web services available which act like a transportation path between the LLA ontology and its specific legacy domain location. The data in the LLA is accessible from the legacy system through the Web Services. The legacy systems are either met office or other information sources with emergency related data. Data from these systems is transported forward with the help of web services to the LLA.

The data transported by the Web services is metadata, dynamically derived from each local information sources providing an abstract description of local data of the LLA. This data forms ontology which reaches the LLA. We endow each LLA in the system with a "Client Ontology Component" (COC) which gives it basic ontology handling capabilities derived from its specific legacy domain. These LLA have their own unique ontology which is made up of a number of concepts. This arrangement works in the following manner:

The LLA starts with a subset its own unique base ontology which is directly a replica of information lying in their specific legacy system domain, which is loaded at startup from an internet resource. They use their local ontology's, handled by the COC, as long as the local knowledge serves for the LLA agent's activity.

When further knowledge is required for instance, an unrecognized term arrives from another agent the COC queries are forwarded to the MA whose work is to find out which agent knows the answer. Before looking for the required agent who knows the answer, the MA holds the part of ontology of every LLA and this part is known as the Global Ontology Component (GOC). The MA looks for the required concept in the GOC of the specific LLA. The MA looks if any of the LLA knows the answer to the queries by going through the GOC of all of them. Once the relevant LLA is found which holds the information of the query, the MA sends a tailored addition of the basic ontology that allows the LLA to continue working. The COC of the LLA stores temporarily the ontology concepts sent to it by the MA but the permanent storage of the newly learnt concept is done in the GOC of the specific LLA's GOC in the MA. Once the TA is found by the MA then the TA helps in concept learning of the other LLA in the following way. Based on the agent model, TA possesses examples regarding a certain concept. They provide learning agents with examples and also they can answer learning agents' questions regarding categorization of a certain example. The supple hierarchical structure of the TA lets it traverse its concept structure from the concept toward its examples and features. This flexibility of structure can help TA to answer the LLA queries based on both concepts, features and examples it holds.

The MA forms a cluster which in real life if thought of can be departments or divisions of an organization or might represent data points in different locations. A group of these MA makes up a whole network or web of connected Manager Agents which provide interoperability and passing of information at the time of emergency and in finding the relevant data sources which hold the data.

Once the information is received by the LLA it is then displayed on the web interface to the required LLA and is also incorporated in the agent's ontology.

## 7. LLA ARCHITECTURE

If we consider a cluster of agents, then each agent has a concept *Ck* and holds examples regarding the concept and describing features in its Component Ontology Component (COC). Also each agent has utilized a learning algorithm to conceptualize *Ck*. In addition to these resources each agent has its unique ontology component residing in the MA known as GOC. Formally, we can represent each agent's COC having four attributes.

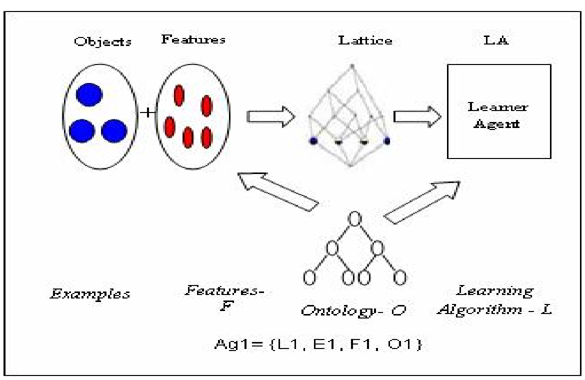


Figure 2 - Ontology Structure in a LLA

### 7.1. Learning algorithm (L)

We use supervised learning algorithms to learn a concept *Ck*. This is the main reason of utilizing a learning algorithm. Also in our framework, a learning agent learns a concept in a supervised manner. This supervision is accomplished by MA. In some cases, when the LLA is not an expert about queried concept but it has some example regarding it, we utilize the LLA by the learning algorithm *L*.

### 7.2. Set of features (F)

To represent a concept, its features play a vital role. We need features to represent concepts, distinguish between them and to describe its corresponding concept. We assume that each individual agent can represent a concept using different features. Having a different ontology means having different set of features for concepts *and* having different hierarchy of relations for them.

### 7.3. Set of examples (E)

In our framework, examples are transferred from teacher to the learner in order to supply the learner with examples of a concept. These examples could be any instances which the teacher agents consider as examples of the queried concept.

### 7.4. Ontology (O)

The concept learning in the multi-agent architecture is carried out in a way that each agent has its own unique ontology and they do not use a common shared ontology. The capability of agents in defining their own individual ontology makes multi-agent concept learning meaningful. Concepts in the ontological hierarchy of our framework are divided into two levels: Well-Understood concepts (WUC) and Vague-Understood concepts (VUC). For each agent, Well-Understood concepts are the concepts which are explicitly present in its ontology structure. Every agent is an expert about its WUCs. Vague-Understood concepts are the concepts that are not explicitly mentioned in the agents' ontology. There are some examples in an agent's repository showing a VUC but the concept is not explicit for the agent.

The underlying principle behind WUC and VUC is that it is not essential for an agent to have every concept explicitly in its ontology; rather it is possible for any agent to have a set of examples with some common features which might not be explicitly formed as a concept in its conceptual hierarchy. In response to a concept query, an agent searches its ontology to find a concept, which is matched by features. In case of success, we consider the agent as a TA and it returns the concept examples to the MA which are directly assigned to the WUC of the LLA's GOC residing at the MA. In case of failure, there is no WUC for the queried concept and the agent is not an expert about the concept, however the agent starts to make a partial lattice to find a VUC and sends back its comprising examples.

At an agent's startup, the COC is responsible for fetching normally from an internet location a base ontology. In order to overcome the limitations of the base ontology, the COC is responsible for accessing the MA for extending its ontology knowledge by accessing the other part of its ontology known as GOC, through the query mechanism discussed later. The results of a query are incorporated by the MA to the global ontology part GOC in the MA for the required LLA and part of it is sent back to the learning agent which required this information, thus extending automatically the ontology as needed. The existence of the other knowledge provider TA is transparent to the LLA which required information, as it directs every query to the COC which further directs the query to the MA.

## 8. AGENTS COLLABORATION EXAMPLE

Let's suppose a scenario. An emergency situation has occurred and disaster management personnel require information of the specific emergency location and other information which might be residing anywhere at the time of emergency. It might be possible that the Disaster management's required information may be residing in another geographical locations Legacy data locations. We suppose LLA Ag1 that is the learner agent which requires some concept to be at a location in Karachi which has come across a query from some other agent residing in Islamabad:

1. An *Agl* has been queried by some other agents about concept *Ck.* The LLA first checks for the required concept in its COC by going through its lattice structure and finding a match for the required query concept.
2. If it fails to answer the query, then it determines that it does not know the concept and should learn it.
3. *Agl* consults the MA of the teacher to look for the agent who will be able to help provide knowledge or answer the query.
4. The MA looks for the required concept in the GOC of the Ag1 from which it received the query. If it does not have a similar concept of the query in the GOC then it goes through the lattice structure of the other Agents GOC's it has to find the agent which holds the concept required by the query.

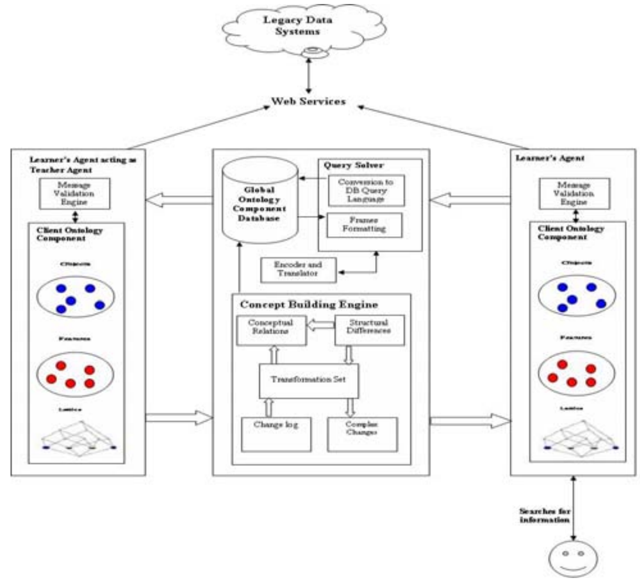


Figure 3 - Low Level Architecture of SEMA

5. Once the MA finds the relevant TA which knows the query answer, then the TA receives the query and uses their learned representations for its own conceptual hierarchy to infer the location of the concept in their ontology.
6. The TA replies to the MA with:
   - "Yes, I know that semantic concept and its location".
   - "I may know that semantic concept" (in case that the Teacher Agent does not have the concept in its ontology but holds only some concepts or features related to the query).
7. Ag2 will send the concept to the MA. Ag2 will act as a Teacher Agent TA and will send examples of concepts to resolve and answer the incoming query of the LLA and send it to the MA.
8. MA will check for the representation format of the answered query and will encode and change the format of the query response to a format which the original LLA which sent the query understands.
9. MA after encoding will send a tailored version of it to the Ag1 where as it will permanently store the newly fetched concept in the GOC of the LLA which it holds.

### 8.1 Query Mechanism

Now the query mechanism from the LLA to the MA and vice versa is presented. It consists of three elements: the Query language, the Query Engine and the Answer Format. One of the simplest query languages is RQL [6]; its syntax is similar to SQL. The Query Engine is responsible for solving the queries made to the ontology. Its performance will be one of the most critical factors in the global performance of the MA, as it could be constantly answering questions coming from LLA.

Query responses are coded in a response format. Once the LLA receives an answer from the MA, it can process its information. This processing - decoding could be costly both for the LLA and for the MA if adequate formats are not chosen. The MA encapsulates the functionality for playing the role of a knowledge provider, storing the ontology encoded, translating interpreting and executing incoming queries, then translating back the results to a format understandable for the LLA. The query engine basically performs these conversions.

## 9. SEMA KNOWLDEGE BASE ONTOLOGY

Ontology describes basic concepts in a domain and defines relations among them. Basic building blocks of ontology design include classes or concepts and properties of each concept describing various attributes [7]. We will be making ontologies by using tool Protégé-Frames. This tool builds and populates ontologies that are frame-based, in accordance with the Open Knowledge Base Connectivity protocol (OKBC) [7]. We can export this format to many extensions like OWL, XML and RDF files.

The sample ontology which we are building is of an Agency Tracking system. This ontology will help agents understand the domain of Agency tracking. This ontology's main task would be to help self-distribute, monitor and coordinate agencies in the disaster situation. A view of the classes in agency tracking is shown in figure 4.

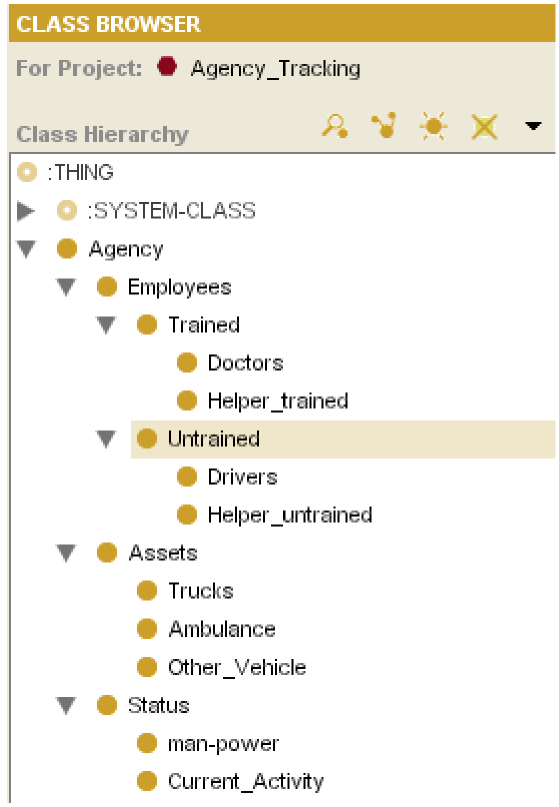


Figure 4. Class view of Agency Tracking System

## 10. CONCLUSION

The importance of effective and integrated disaster management system cannot be denied. Numerous steps taken by government towards its development can be seen. SEMA is the solution in which data sources and services are made available through web services described by ontologies, communicated through agents allowing interoperability and reasoning to create ample response adapted to user objectives. This paper does not cover all the issues which are needed to be researched upon in future.

## ACKNOWLEDGMENT

We are really thankful to Allah Almighty for giving the faith within us to accomplish this task. Many people helped to make this activity possible, especially a bundle of thanks to our teachers and classmates who were there for us when we needed them.

## REFERENCES

[1] Walter, J., *World Disasters Report* 2003. International Federation of Red Cross and Red Crescent Societies, Geneva. See Chapter 8: Disaster data: key trends and statistics.

[2] Walter, J., *World Disasters Report* 2006. International Federation of Red Cross and Red Crescent Societies, December 2006. See Facts and Figures.

[3] Zaidi, S., *Microsoft offers technological assistance in quake-affected areas*. The News, 21 December 2005. <http://www.apnic.net/mailing-lists/s-asia-it/archive/2005/10/msg00052.html>

[4] Asian Earthquake OCT 2005. <http://www.reliefsource.org/foss/index.php/Use:Asian-Earthquake-OCT-2005>

[5] Dominguez, J., Cabral, L., Hakimpour, F., Sell, D., Motta. *E.: IRS-III: A Platform and Infrastructure for Creating WSMO-based Semantic Web Services*. In proceedings of the Workshop on WSMO Implementations, CEUR Workshop Proceedings, Vol. 113. Frankfurt, Germany (2004).

[6] G. Karvounarakis, S. Alexaki, V. Christophides, D. Plexousakis, and M. Scholl. *RQL: A declarative query language for RDF*. In The 11th Intl. World Wide Web Conference (WWW2002) <http://citeseer.nj.nec.com/556066.html, 2002>

[7] Eliza Sachs, Getting Started with Protege-Frames <http://protege.stanford.edu/doc/users.html>