Knowledge Evolution System for Dynamic Emergency Planning and Response

Saravanan Muthaiyah
Multimedia University Malaysia
saravanan.muthaiyah@mmu.edu.my

Murali Raman
Multimedia University Malaysia
murali.raman@mmu.edu.my

Magiswary Dorasamy
Multimedia University Malaysia
magiswary.dorasamy@mmu.edu.my

Abstract

The knowledge requirements of managing emergencies are inherently dynamic and evolving in nature. Existing KM systems in the context of emergency management are essentially static. This paper suggests the use of multi-agent systems for implementation of a more reliable and dynamic emergency planning and response system that can support the evolutionary nature of knowledge in emergencies. To validate this approach and to provide proof-of-concept, several tools have been used in this paper such as Jena and Protégé. We demonstrate how multi-agents are implemented for knowledge evolution and change management using the JADE platform.

1. Introduction

Emergency planning and response efforts require up-to-date and dynamic information. Integration of knowledge management and the Semantic Web helps provide dynamic emergency data for swift emergency planning and response. At present most knowledge management systems are not equipped with a

This echoes the view of March and Simon (1958) who state that successful organizations are able to adapt to any dynamic environment [3]. The information processing theory states that the role of having accurate and up-to-date information is vital particularly when organizations deal with turbulent environments [2]. Implementation of a knowledge management system that can support managers to proactively respond to a highly turbulent environment will benefit an organization [2]. This would include organizations that plan and prepare for an emergency situation, either man-made or due to the forces of nature [4].

Integration of knowledge management with Semantic Web would promise the dynamism of data required to support the evolving nature of knowledge within emergencies. Berners-Lee defines the Semantic Web as "a web of data that is directly or indirectly processed by agent systems." [5]. The World Wide Web technology today is based on HTML technology. It is only useful for presenting static HTML data and is not machine processable data. This limitation is overcome by the advent of Semantic Web technology. framework to support such requirements. A system which is dynamic as apposed to static can address knowledge evolution within the emergency planning domain. This paper applies a multi-agent system (MAS) for deploying emergency related data using the Semantic Web.

2. Knowledge Management and Semantic Web

Knowledge management encompasses a wide range of disciplines [1]. Groupware, decision support systems, expert systems and other forms of collaborative systems are examples of technology related to knowledge management [1]. An organization's ability to survive given dynamic changes within its environment is contingent upon its ability to quickly respond to change [2]. This includes among others the ability to effectively manage its knowledge resources [2]. Burnell et al. assert that "an effective knowledge-based organization is one that correctly captures, shares, applies and maintains it knowledge resources to achieve its goals" (p.203). Our approach is based on Semantic Web technology as it is the platform recommended for agent systems which allows dynamic processing of emergency data [5]. This will also support our cause of implementing systems that capture knowledge evolution [6][7].

Semantic Web technology helps promote and create meaningful content out of Semantic Web data, as it is machine understandable. It is built on the environment where software agents roam and analyze emergency related data to support faster decision making. Semantic Web technologies utilize ontologies and taxonomies to process and interpret web contents [8][10][11][12]. Therefore, it is imperative to capture dynamically all ontology-based emergency data via an ontology based Knowledge Management System (KMS) [9] for all emergencies. Research in the area of ontology engineering has reached a considerable level of maturity and stability. With the advent of editing tools such as Protégé and knowledge representation models, this progress has grown exponentially. However, very little work has been done for the evolutionary aspects of ontologies, particularly in the domain of emergency response. The Semantic Web is a

IEEE computer society

978-0-7695-3869-3/10 $26.00

distributed and collaborative environment where ontologies will naturally evolve and co-evolve as such research on knowledge evolution becomes crucial. A lot of work has been done in the area of evolution of single ontologies, but very little has been done for the evolution of shared inter-organizational ontologies.

3. Shared Knowledge Repository and Emergency Systems

In order to facilitate knowledge reuse and in the same time to allow experts in the emergency domain to input their knowledge, we propose a hierarchical repository structure. Knowledge will be organized at different levels, each emergency ontology inheriting the knowledge from one or more parent ontologies. Ontologies will completely inherit the knowledge from its parent ontologies (e.g. no partial inheritance), so for multiple inheritance relationships it is very important that the knowledge inherited from parents to be consistent (i.e. no naming clashes).

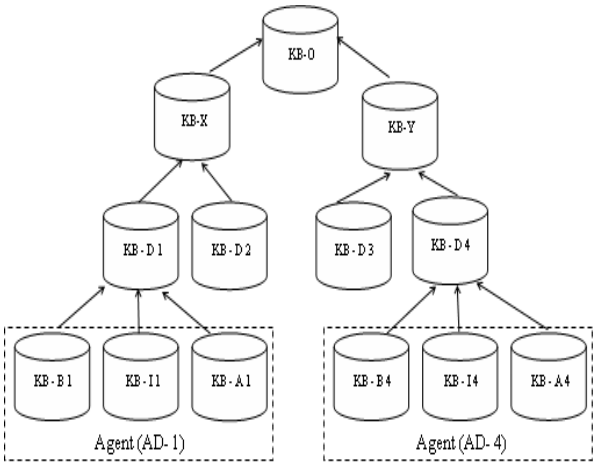

Figure 1: Shared Knowledge Repository

Figure 1 illustrates a hierarchical shared knowledge repository. There are two agents, (A-D1and A-D4). The agents have their respective locally developed emergency ontologies and also inherit parts of the shared emergency knowledge repository. A-D1 has three local ontologies that inherit emergency data from KB-D1 for KB-B1,KB KB-I1 and KB-A1. In the context of emergencies this can be represented as emergency data on fire (e.g. KB-B1) and flood (e.g. KB-BI1). KB-D1 inherits emergency data from KB-x and KB-0. Agent A-D4 has local ontologies (KB-B4, KB-I4 and KB-A4) that incorporate knowledge from KB-D4. KB-D4 in turn inherits emergency data from KB-Y and KB-0. When agent A-D1 and A-D4 collaborate, they know for sure that at least part of their knowledge is similar (e.g. KB-x and KB-0). The reuse of knowledge is much easier using the shared hierarchy as emergency data definitions can be easily inherited from existing knowledge. Also, different contradictory pieces of knowledge can be encoded in different ontologies which are not in an inheriting relation (e.g. one is not a parent of the other). At the same time, the communication between agents is also easier this way since they share common knowledge, so at least partially, they "speak" the same language.

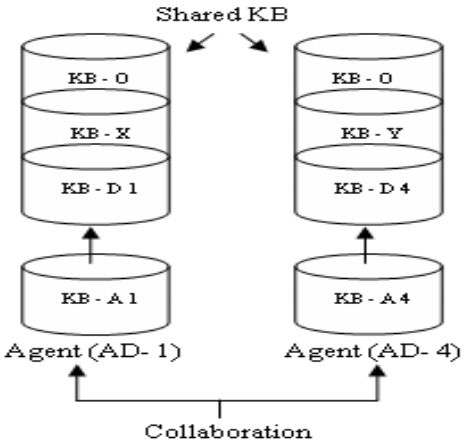

Figure 2: Shared and Distributed Emergency Knowledge Repository

A hierarchical repository structure would help with the knowledge reuse, but nowadays is not realistic to consider that all the developed ontologies are under a central control and available at all times to everybody. As such, a distributed model of the hierarchical repository is would be more appropriate. Each ontology will have its own creator/maintainer and will be able to inherit knowledge from the ontologies created by others. For instance KB-A1 and KB-A4 in figure 2 above are really local ontologies that maintain data on specific emergency type definitions that are highly rich for one specific emergency or disaster such as wild bush fire. In the higher level KB fire is expressed more generally e.g. forest fire and fire caused by an explosion.

4. Knowledge Evolution

An important part of maintaining emergency data definitions let it be for flood, hurricanes, tornadoes or even tsunamis is the problem of how to manage the evolution of the individual domain ontologies for the different disasters. We introduce an approach where

first, the knowledge is sliced into an intermediate ontology format. This step can be performed by various plug-ins. There is a need for an intermediate ontology format since the destination frame-like knowledge representation might be quite restrictive and implementing a translation engine into each plug-in would duplicate a lot of work. This way we'll have the same translation engine performing the work on knowledge extracted from any known source. It will deal with constructing a stand-alone ontology which captures as much as possible knowledge from the intermediate ontology file into the knowledge representation of the destination. For this paper we would only limit the experiment to fire emergencies.

We introduce a versioning mechanism which enables sharing of static ontology versions (see table 1). This way, when knowledge is inherited, it's actually the knowledge from a certain version (snapshot) of that ontology, which doesn't change over time. The maintainers can update their ontologies locally and release new of versions when they are ready. At the same time, an ontology maintainer can decide to upgrade the inherited parent ontologies to newer versions. Suppose the direct parents of KB-D1 (v1) are KB-X (v1) and KB-Y (v1) and if the maintainer of the KB-D1 decides to upgrade one of his parent ontologies, (i.e. KB-X (v1) or KB-Y (v1)) he has to upgrade it to inherit from other newer parent ontologies as well, due to version dependencies.

The maintainer will update to inherit knowledge from KB-X (v2) as it still uses KB-0 (v1) but cannot update to inherit knowledge from KB-Y (v2), as it inherits from KB-0 (v2). Also there are no versions of KB-X that inherit knowledge from KB-0 (v2). Given that only one version of the ontology can exist at any one time, inherited knowledge (i.e. KB-0 (v1) and (v2)) cannot be inherited at the same time, directly or indirectly, by the same ontology. This also applies to KB-D4 as it uses KB-0 (v2) but KB-X update will be based on KB-0 (v1).

Table 1 : Ontology Versions

Ontology	Versions (Inherit from)
KB-0	v1, v2
KB-X	v1 (KB-0 v1), v2 (KB-0 v1)
KB-Y	v1 (KB-0 v1), v2 (KB-0 v2)
KB-D1	v1 (KB-X v1, KB-Y v1)
KB-D4	v1 (KB-Y v1, KB-X v1)

After selecting the new versions of ontologies to inherit from, we put the inherited knowledge together. When attempting to upgrade to newer versions of inherited ontologies, naming clashes can happen as more than one version could have defined the same ontological concept. This would make both ontologies "incompatible" and impossible to inherit from both versions at the same time.

5. Semantic Mapping for Dynamic Matching

The semantic mapping will relate data schemas and is responsible for formally capturing the meaning of the data by referring to the agreed-upon business terminology. By creating a knowledge base for this we can analyze user preferences based on user behavior. Table 2 below, describes in detail the components of the proposed architecture.

Table 2: Components and definitions of proposed architecture

Components	Details and Functions
User	Agent system or domain expert
Disaster Module	Disaster monitoring module
Match agent	Matches disaster events with real-time applications
Pattern recognition	Analysis of disaster events which enables mining for data inferences
Rule parser	Parsing of rule based on needs of knowledge base
Knowledge base	Creation of knowledge base for all rules parsed
Rule generator	API based rule generated for checking consistency of rules
Rule ontology	Ontology rule for each data label classification
Knowledge of disaster and recovery	Knowledge base built on ontology and mapping
Information for user	Knowledge base output for feedback to domain expert

6. Proposed Architecture

We propose the MAS framework in Figure 4 to handle the ontology knowledge evolution process (e.g. upgrade a local ontology to work with a new version of a shared ontology). Figure 3 depicts the upgrade operation. The Ontology Agent (OA) is requested by the other agents to get access to the respective disaster domain ontologies such as flood, tsunami, earthquake, tornadoes and hurricanes. However in this implementation we have only implemented it for a fire emergency. OA uses OWL ontologies and stores the ontology modifications in an evolution log.

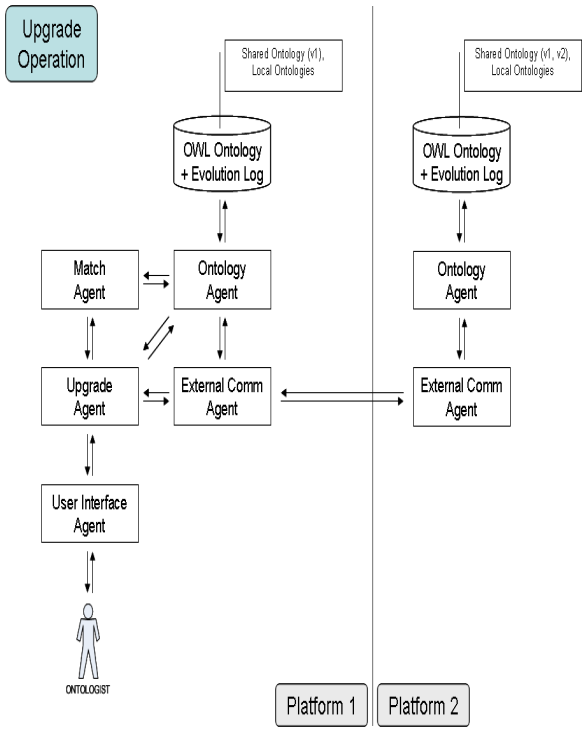

Figure 3: Upgrade Operation

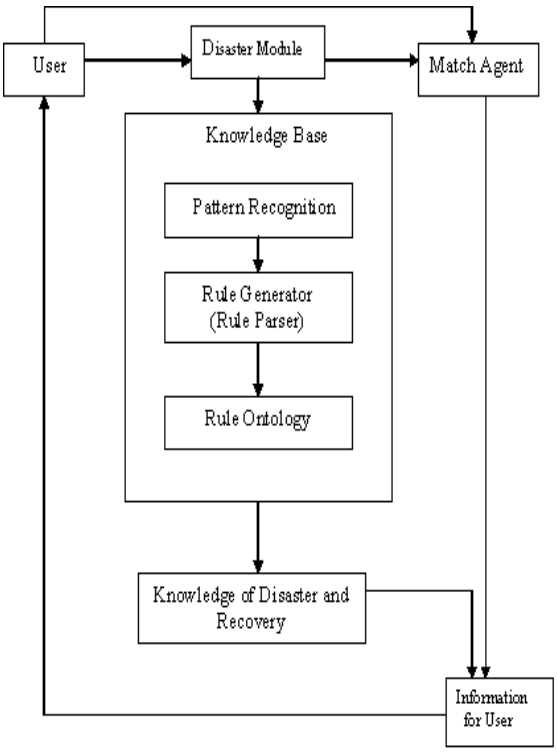

Figure 4: Integration Architecture

I. New elements

The only problem introduced by new elements in a new version of an ontology is if their names/ids collide with elements locally defined somewhere else. If the multiple definitions are inherited, there is nothing that can be done; the inherited knowledge is "incompatible". In case the inherited knowledge collide with the local knowledge, there are two possibilities: the two elements are semantically different so the only option is to have the local element renamed since we cannot alter the inherited element; the other case is when a local element was added later to a newer version of the inherited knowledge so we don't need to define it locally anymore, since it's inherited from the new version.

II. Deleted elements

If inherited knowledge elements were used locally and deleted from the parent ontologies, a straight forward solution is to recreate locally the same piece of knowledge; another approach is to treat the operation the same way as if a local element is deleted and to adapt the local ontology.

III. Modified elements

This is a complicated situation and can be treated the same way modifications to local elements are treated, as if the modified piece of knowledge would be local. Imagine when there is limited inherited knowledge available and two or more ontologies start to develop the same kind of knowledge. One can suggest a piece of knowledge to be incorporated in an upper ontology and to become inherited knowledge for the others; a "sliced piece" of knowledge can be suggested to the maintainer of the upper ontology to be incorporated [17]; the maintainer may accept it, integrate (import) it and release a new version of the ontology. After this, the other ontologies that inherit knowledge from the original version can be upgraded.

An existing knowledge repository might need to be augmented with knowledge from other existing monolithic ontologies or even with knowledge from different systems using a different knowledge representation model. In the latter case, an extra step is required to perform a knowledge translation operation into your own knowledge representation space. One way to start importing knowledge is to build a list of terms that you want to import knowledge about (for example things like "government" and "weapon"). In general, knowledge elements are highly connected

through semantic links; one can build an algorithm to extract the desired terms and related knowledge by "slicing" it, by following the links as far as a set maximum depth [2].

6. Implementation Tools

JADE (Java Agent DEvelopment Framework) [14] is anagent framework that we have used to develop our agent-based application. Figure 5 shows the agents that we had used to deploy the process described in figure 4. JADE creates an environment for agents to communicate and is implemented in Java. It is compliant with the Foundation for Intelligent Physical Agents (FIPA) specifications and is capable of facilitating multiplatform agent development. As such JADE we chose JADE as the development environment for this project. Agents produced in Java using the JADE environment have the distinct advantage of almost always being run-able with minimal supporting download.

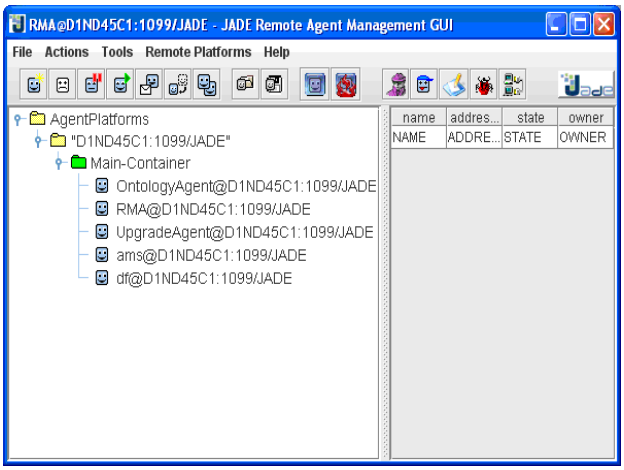

Figure 5: Agents loading in JADE

Jena [15] is a Java framework for building Semantic Web applications. It provides an API for working with RDF, RDFS, OWL ontologies and SPARQL queries and also includes a rule-based inference engine. Jena was initially developed by HP and released as open source later on.

7. Conclusion

This paper highlights the process of updating a constantly evolving knowledgebase in the domain of fire emergencies. A multi-agent system (MAS) environment and a framework has been proposed for multiple agents to deploy changes in a shared ontology environment such as update, deletion and renaming of classes in a dynamic environment. The upgrade operation is dependent upon a Semantic mediation technique that we have developed earlier using the Semantic Relatedness Score (SRS) [16]. The advantage of SRS compared to other measures is that it is based on a hybrid model that combines both semantic and syntactic matching. It has been empirically tested and produced a 92% accuracy for matched results [16]. Our proposed mediation process is a mixed-initiative effort that involves intelligent agents and domain experts. We have implemented our work to support a local government agency here in Malaysia i.e. the Malaysian Association of Social Workers (MASW). The agency is currently using a Wiki-based system that is static.

References

[1] J. D.Gupta, and S. K.Sharma, "Creating Knowledge Based Organizations", IDEA Group Publishing, 2004.

[2] L Burnell,., J. Priest, , and J. Durrett, "Developing and Maintaining Knowledge Management System for Dynamic, Complex Domains. In J. Gupta & S. Sharma (Eds.), Creating Knowledge Based Organizations. London: IGP., 2004.

[3] March, J. G., & Simon, H. A. (1958). *Organizations*. New York: John Wiley & Sons.

[4] Kostman, J. T. (2004). 20 Rules for Effective Communication in a Crisis. Disaster Recovery Journal., 17(2), 20.

[5] T. Berners-Lee, J. Hendler, and O. Lassila,, "The Semantic Web: A New Form of Web Content That Is Meaningful to Computers Will Unleash a Revolution of New Possibilities," *Scientific Am.,* vol. 284, no. 5, May 2001, pp 28-37.

[6] R. Howard, and L. Kerschberg, "Brokering Semantic Web Services via Intelligent Middleware Agents within a Knowledge-Based Framework", presented at Intelligent Agent Technology (IAT 2004), Beijing, China, 2004.

[7] R. Howard and L. Kerschberg, "A Framework for Dynamic Semantic Web Services Management," Special Issue on Service Oriented Modeling, International Journal of Cooperative Information Systems, Volume 13, December 2004.

[8] A. Barbir,"Web Services Security: An Enabler of Semantic Web Services", Nortel Networks, Canada, 2002, pp. 1-5.

[9] S. Muthaiyah, and L. Kerschberg, "A Hybrid Ontology Mediation Approach for the Semantic Web", International Journal of E-Business Research Volume 4 Issue 4, USA, 2008, pp. 79-91.

[10] S. Luke et al., "Ontology-Based Web Agents," Proc. First Int'l Conf. Autonomous Agents, ACM Press, New York, 1997, pp. 59–66.

[11] D. Fensel, "The Semantic Web and Its Languages," IEEE Intelligent Systems, vol. 15, no. 6, Nov./Dec. 2000, p. 67–73.

[12] D. Fensel et al., "OIL in a Nutshell," Proc. European Knowledge Acquisition Conference (EKAW 2000), R. Dieng et al., eds., Lecture Notes in Artificial Intelligence, no. 1937, Springer-Verlag, Berlin, 2000, pp. 1–16.

[13] J. Ohler, "Web 3.0 – The Semantic Web Cometh: What Happens When the Read-Write Web Begins to Think?", Retrieved on June 15, 2009 from http://www.jasonohler.com/pdfs/Web3-SemanticWebCometh.pdf

[14] Java Agent DEvelopment Framework, Retrieved on June 15, 2009 from http://jade.tilab.com

[15] Jena – A Semantic Web Framework for Java. Retrieved on June 15, 2009 from http://jena.sourceforge.net

[16] Muthaiyah, S., Barbulescu, M., and Kerschberg, L "Managing Change and Evolution via Hybrid Matching", *WSEAS Transactions on Computers*, Volume 7, Issue 10, pg 1700-1710, October, 2008, ISSN: 1109-2750

[17] Chaudhri, V. K., Stickel, M. E., Thomere, J. F. and Waldinger, R. J., "*Using Prior Knowledge: Problems and Solutions*", Proc. of the 17th National Conference on Artificial Intelligence and the 12th Conference on Innovative Applications of Artificial Intelligence, 436-442. Austin, Texas: AAAI Press/The MIT Press, 2000.