Software Agent Oriented Information Integration System in Semantic Web

Pa Pa Nyunt, Ni Lar Thein
University of Computer Studies, Yangon, Myanmar
ucsy23@most.gov.mm, nilarthein@mptmail.net.mm

Abstract

Developing an information-sharing capability across distributed heterogeneous data sources remains a significant challenge. Ontology based approach to this problem show promise by resolving heterogeneity, if the participating data owners agree to use a common ontology (i.e., a set of common attributes). Such common ontology offers the capability to work with distributed data as if it were located in a central repository. This information sharing may be achieved by determining the intersection of similar concepts from across various heterogeneous systems. However, if information is sought from a subset of the participating data sources, there may be concepts common to the subset that are not included in the full common ontology, and therefore are unavailable for information sharing. One way to solve this problem is to construct a series of ontology, one for each possible combination of data sources. In this way, no concepts are lost, but the number of possible subsets is prohibitively large.

This paper describes a software agent oriented information integration system that demonstrates a flexible and dynamic approach for the fusion of data across combinations of participating heterogeneous data sources to maximize information sharing. The software agent generates the largest intersection of shared data across any selected subset of data sources. This ontology-based agent approach maximizes information sharing by dynamically generating common ontology over the data sources of interest.

Our country, Myanmar, have seven states. Each state has one agricultural research center. And so, our approach was validated using data provided by seven (disparate) agricultural research centers by defining a local ontology for each research center (i.e., data source). The Ministry of Agriculture & Irrigation (MOAI) manages the all states’ agricultural research centers, each of which has evolved a variety of agricultural models for managing research proposals over the past decade. Because of the historical nature of these evolutions, both the agricultural models and their associated (heterogeneous) data collections are deeply rooted. A system was needed that could merge data from the heterogeneous systems as if the data was gathered and stored in a centralized repository.

In our approach, the ontology is used to specify how to format the data using XML to make it suitable for query. Software agents provide the ability from the data sources to dynamically form local ontology for each research center. By using ontology based software agent, the cost of developing this ontology is reduced while providing the broadest possible access to available data sources.

Keyword: Ontology, Extensive Markup Language (XML), heterogeneity, Graphical User Interface (GUI)

1. Introduction

In Ministry of Agriculture & Irrigation (MOAI), each research center has a little various ways to collect research proposal information. Each collection of information and its corresponding ontology are depending on a unique agricultural model. Moving to a single common ontology by resolving heterogeneity of data across all research centers will result in an intersection of all data concepts. Unfortunately, only common concepts are retained. Moreover, if two or more research centers have a concept which is not common, then valuable information is not being represented. In this case, a common ontology that results in loss of such data has an unacceptable impact on the associated agricultural models.

There are many unfortunate problems by using only ontology. Concepts can be lost in a common ontology. Figure 1 gives an example of how concepts can be lost in a common ontology. Consider three data sets A, B, and C each of which contain four concepts, Center Name, Model Name, Usage and one other that is not common. Let’s examine the merged data from data set A and B. The first two concepts match across all four but the fourth concept is lost. This is unfortunate because we are concerned only about data from sets A and B (see figure 1). So a better method is needed to overcome this problem.

In this case, the new approach needs to add Duration to the ontology when only data sets A and B are being considered. If this is possible, then full visibility of all data across all combinations all participating data sets could be accomplished with a series of merged ontology. The ontologies are merged in a way that provides for each possible combination of source data concepts, as compared to a single common ontology. On the other hand, there are a large number of possible combinations a user may choose potentially generating a huge number of ontology combinations. Current ontological approaches for merging heterogeneous data sets have been successful, but
require all data owners to participate in building a single common view.

![Diagram of ontologies and data sets](image)

**Figure 1. Common and subset ontologies between disparate data sets**

For this reason, a large number of possible data set combinations ontology for every possibility cannot be "pre-built". Instead, we support to use software agents to build the desired ontology combination on-the-fly for each user-generated merged and query/search operation. Seven disparate data sets have disparate ontologies and agents are used to integrate these various ontologies. These agents are gathering by agent community and then arrive to the unified view of data set. In this way, only relevant merged are implemented, avoiding the need to generate all possible combinations while satisfying the possibility for merges that consider all possible combinations.

2. Related Work and Problem Issues

Web services is one of the main forces driving the internet from its infant, a vast collection of text and images, to today’s huge growing marketplace of service providers and consumers. Agent technology plays an important role in this evolution. One of the fundamental problems in building open Web-based applications is how to find information providers and how to integrate information agents in such a dynamic environment. There is an obvious need for a standardized meaningful communication among agents to enable them to know each other and perform collaborative task execution.

The first approach relies on ontology based language for agent services description. This language exploits ontology of service domain, and provides the flexibility for developers to plug in a suitable language to describe the constraints. Multiple matchmaking strategies based on agents’ service ontology are given to help agents finding appropriate service providers. The series of strategies consider various features of services providers, the nature of requirements, and more importantly the relationships among services.

The second approach tries to manage patients in a shared-care context is a knowledge-intensive activity. This approach supports cooperative work in medical care and to enhance their ability of interacting with each other with computational resources. A major shift is needed from centralized first generation health-care information systems to distributed environments composed of several interconnected agents, cooperating in maintaining a full track of the patient clinical history and supporting health-care providers in all the phases of the patient-management process. This approach outlines a general methodology to make architectural choices while designing or integrating new software components into a distributed health-care information system. A particular stress is laid on the specification of shared conceptual models, or ontologies, providing agents committing to them with the common semantic foundation required for effective interoperability.

The third approach to path finding in the intelligent graphs, with vertices being intelligent agents. This approach is based on logical inference in distributed frame hierarchy. Presented approach can be used for implementing distributed intelligent information systems that include automatic navigation and path generation in hypertext. The merging approach of creating a unified source is not scalable and is costly. Besides, an integrated information source would need to be updated as soon as any information in any individual source changes. [9]

The fourth approach to solve this problem. A system that enables interoperation among information sources using ontologies needs to resolve the terminological differences between ontologies. In this work, we present several methods that we have implemented two methods based on linguistic similarities of words based on word similarity computed from a domain-specific corpus of documents. This approach discusses a method that uses both heuristics produces good results. [8]

Our approach is based on the first idea. It was validated using data provided by seven (disparate) agricultural research centers by defining a local ontology for each center. In this experiment, the ontologies are used to specify how to format the data using XML to make it suitable for query. Consequently, software agents are empowered to provide the ability to dynamically form local ontologies from the data sources. In this way, the cost of developing these ontologies is reduced while providing the broadest possible access to available data sources.

3. Theoretical Foundations

The word ontology has been taken from Philosophy, where it means a systematic explanation of the Existence. In the Artificial Intelligent field, first Neches and colleagues defined ontology as follows “ontology defines the basic terms and relations comprising the vocabulary of a topic area as well as the rules for combining terms and relations to define extension to the vocabulary”. Later, in 1993, Gruber’s definition becomes famous “an ontology is an explicit specification of a conceptualization”, being this definition the most referenced in the literature. In 1997, Borst slightly modify Gruber’s definition saying that:
“Ontologies are defined as a formal specification of a shared conceptualization”[4]. Other definitions are: “ontology is a hierarchically structured set of terms for describing a domain that can be used as a skeletal foundation for a knowledge base” and “ontology provides the means for describing explicitly the conceptualization behind knowledge represented in a knowledge base.”[4]

Agent can be defined to be autonomous, problem solving computational entities capable of effective operation in dynamic and open environments. Agents are often deployed in environment in which they interact, and may be cooperate; with other agent (including both people and software) they have possibly conflicting aims. Agent can be act as a design metaphor, a source of technologies as design, etc.

In the knowledge management area, an increasing number of companies are realizing that their own intranets are valuable repositories of cooperate information, but without understanding of how to apply it effectively this information is likely to be useless. Knowledge management is concerned with the acquisition, maintenance and evaluation of the knowledge of an organization. Information agent typically have access to multiple heterogeneous and geographically distributed information sources, in the internet and cooperate intranets and search for relevant information, on behalf of their users or other agents. This includes retrieving, analyzing from geographically distributed, distinct autonomous information sources. The agent should be able to present both unified view and different perspectives of the information to the user. This process will involve fusing heterogeneous data.

These various ontology based approaches[4,7] provide methods to discover the homogeneity that may be found among heterogeneous data sets, and from that, build a common ontology. However, these approaches assume that participants are capable of migrating their data to a new ontology; this was not the case in our problem. The varied data collections within MOAI are tightly coupled to their associated business processes. So much so, that it became unclear whether the agricultural model drives the data or the data drives the agricultural model. Migrating the data to a common ontology would necessitate a prohibitively expensive change to long and well-established agricultural models. Thus, these powerful approaches are not suited to our problem without a variation from previously seen ontology-based data fusion.

4. Proposed Approach

Information are merged by using ontologies is great. Current ontological approaches to the merging of heterogeneous data have been successful, but require the data owners to adapt a single common ontology because of the importance of concepts occurring in most, but not all data sources. So the solution is the series of ontologies for all possible combinations of the underlying data sets. However, in our approach, this is an intractable problem and we decided to explore the possibly of using software agents to perform the ontology building tasks automatically.

In our approach, Extensible Markup Language (XML) is the foundation for capturing data. Each research center has a different mechanism for capturing data, from databases, to spreadsheets, to ad hoc text files and various combinations. The various data formats are converted into XML by using a series of specified software tools. XML provided an efficient mechanism to bring the distributed and heterogeneous data formats into powerful, flexible and common formats thereby providing a standard ontology to software agent interface. Consequently, abstracting unnecessary details from the underlying data sets, which provide to be a sufficiently rich environment for the software agents to perform the merging of ontologies.

And then, the coordinator for the system was designed. These mechanisms allow software agents to understand the various data sets and enable the agents to merge ontologies. Even though formal ontologies have a great many strengths, there are many relations associated with using them. In our problem, the data owners who understood the data did not have the time for learning these relations. To address this conflict, we devised a simple system that specifies data concepts by defining the equivalence relations between a data owner’s local data concepts and other participating data concepts.

For example, as shown in Figure 2, the first data set, Data Owner A provides a simple list of data elements. In a parallel column, the list of elements is repeated. The first column is labeled Local Data Concept List; the second column is labeled Master Data Concept List. Together they represent a simple ontology, i.e.; a specification of the data concepts represented by a mapping from the Local Data Concept List to the Master Data Concept List.

Next we incorporate data from Data Owner B where data owners A and B know each other’s data. This is the case across most of the MOAI system; most data owners know their data as well as its relationship to
a small number of the other data systems. This knowledge context provided the basis for constructing composite ontologies (i.e., master data concept).

For the second data set, Data Owner B looked at Data Owner A’s ontology mapping. Data Owner B could then provide us the knowledge needed to build a new ontology mapping, mapping B’s data to the Master Data Concept List. Each data concept in a local data list was mapped to the same concept in the Master Data Concept List. If Data Owner B had data that was not in the Master Data Concept List, a new entry was added to the Master Data Concept List.

Conversely, if a data concept in the Master Data Concept List was not present in the local ontology then there was no mapping established from his Local Data Concept List to the Master Data Concept List. In the end, the system consists of three elements (1) Master Data Concepts List, (2) Local Data Concept List to Master Data Concept List Ontology for Data Set A, and (3) Local Data Concept List to Master Data Concept List Ontology for Data Set B.

This process continues for each new data set. Each new data owner uses the previous work to help determine their ontology as a specification of the mapping from their Local Data Concepts List to the Master Data Concepts List. Questions pertaining to proper mapping are resolved (through discussion) between the new data owner and the data owner that previously added the concept to the Master Data Concepts List.

Thus, in this process, the Master Data Concepts List is a union of the data concepts across all participating data sets, and a given data set’s ontology is a mapping specifying the relationship between the intersection of that data set’s local data concepts and the master data concepts. Relationship among a selection of the local data sets’ ontologies can be determined using the Master Data Concepts List as a point of common reference. It is interesting to note that there is no centralized ontology for the entire system.

Instead, it is distributed across the ontology mappings of the individual data sets and the Master Data Concepts List. For example, Data Set A’s “Model Name” specifies the same concept as Data Set B’s “M Name” but this cannot be directly determined at one centralized point; rather it is determined via the data set ontologies and the Master Data Concepts List. Software agents use this distributed ontology to provide the functionality of a centralized ontology along with the ability to be flexible in meeting the varied needs of the users.

As described above, we applied this approach on data from seven disparate agricultural research centers. These centers are large and present massive data sets across a diverse set of repositories (e.g.; databases, spreadsheets, and simple ASCII files, etc.). We manually created the local ontologies, and used these ontologies to create XML representations of the data at the research. From this base, we then applied our approach to building dynamic on-the-fly ontologies using agents.

5. Types of agents and ontology based agent’s community

We establish a software agent that is assigned the responsibility of knowing how to retrieve data from the underlying data repository as well as present the data to the overall system for merging. The data agent uses the ontology that was built by the data owner for that local data set. That ontology mapping allows the data agent to act in a bilingual manner, that is, to understanding both the local and master concepts. Based on the master data concepts side of the ontology, the agent understands the language of the agent community; based on the local data concepts side of the ontology, the agent understands how to retrieve his local data; and based on the mapping between the two sides of the ontology, he understands how to translate between the two.

A Data Integration Agent is the final piece of the system that builds merged ontologies, on-the-fly, according to the user’s desires. The Data Integration Agent directs and coordinates the activities of the data agents in the system’s software agent community. The Data Integration Agent is also responsible for accepting requests from a Graphical User Interface (GUI) Agent, another important type of bilingual agent, responsible for translating user requests into the language of the software agent community, and then translating the results into visualization useful to the user. The Data Integration Agent distributes the GUI Agent’s requests to the appropriate data agents, merges the results from the data agents, and passes the merged data back to the GUI Agent.

Because this is a software agent community, the modules of the system, in the form of software agents, are very loosely coupled. This is true (first) because each agent can be built in any manner, irrespective of the other agents but complaint to the agent community’s web-based interface. Second, the agents may be geographically separated, operating from any accessible points on the web. Third, there is fault tolerance in the system; the system will continue to function if some data agents are unavailable. A missing agent obviously cannot contribute data to a solution, but will not prevent a solution from being created. Fourth, there may be multiple GUI agents to meet the user’s variety of preferences from the user community. Clearly, the agents and ontology approach provides an eloquent efficient and extensible solution.

6. Constructing common ontology by using software agents

The Graphical User Interface (GUI) Agent asks the Data Integration Agent for a list of available data agent. The Data Integration Agent then checks the list of data agents that are registered and verifies the availability of each. This agent then reports the
availability to the GUI Agent who displays the available
data sources to the user. Then, the user selects the
desired data sources and the software agents
dynamically create a merged ontology for the selected
data sources.

To create this merged ontology, the Data Integration Agent sequentially distributes the Master Data Concepts List to the data agent chosen by the user. The first data agent compares this concept list to his local ontology and deletes the data concepts that are not in the local ontology. Then data agents hands the reduced data concept list back to the Data Integration Agent who passes the reduced list to the next data agent selected by the user. This process continues for each of the user-selected agents until all have seen the list.

The remaining data concepts list resulting from this process is the intersection of the data concepts captured within the participating system. In building the Master Data Concept List, the data owners each added concepts from their local data sets that are new concepts to the Master Data Concepts List. The data agents remove data concepts that are not part of their local ontology from a copy of the Master Data Concept List.

The final reduced data concept list constitutes a shared ontology across the participating data sets which is dynamically generated based on a user’s request and is evaluated against the latest information from each local data source. Participating agents can each understand and provide information about all the data concepts which significantly increase the capability of current ontologies.

7. Extracting information from merged data

The user interface is made by GUI to specify a query over merged data. GUI gives the user of high quality querying capabilities. Querying is passed to the Data Integration Agents who partitions the query out to the selected set of data agents. Agents are then responsible to fulfill the query over their local data set. The underlying XML data set may be stored in any manner usable by the data agent. Each data agent uses the local ontology mapping to convert the query request from the Data Integration Agent into a format usable against the respective data set. The answer to the query is then translated back into the format used by the software agent community and returned to the Data Integration Agent. The Data Integration Agent then passes the assembled query response back to the GUI agent for presentation to the user.

8. Results

Seven disparate agricultural research center produces \( (2^7 - 1) \) or 127 possible combinations of merged data sets over which a user might wish to query. This number is too large for the data owners to build all 127 possible ontologies. Yet large enough to test the strength of our software agent enabled ontology building approach.

The data owners were showing great excitement and interest about the potential solution using software agents as described. In fact, all prior efforts with the same goal were unsuccessful. These data owners are extremely busy; they could only give small amounts of time to help validate our approach. Indeed, the approach proceeded better than expected. It took more time to explain the approach to the data owners than to actually incorporate their data. Some data took longer than others to incorporate (i.e.; define in terms of a complaint ontology) because varied native formats, but none of the data took more than a couple of days.

The system performed without a glitch. Acquisition times for query results were negligible, with network latency being the bottleneck. Delays were similar to downloading a web page of typical complexity, well within most typical users’ to tolerances for delay. The sponsor deemed the prototype a success. Due to the success of the prototype, the MOAI decided to implement the system across all of its installations.

9. Conclusions and future work

We have described a problem domain where data owners have data that they wish to share, but they cannot move to a single common ontology because of the potential loss of information (incoherency). One inadequate approach uses a series of ontologies for all possible combinations of data, but is prohibitively expensive.

Using our approach we have demonstrated the use of software agents to dynamically create merged ontologies, while providing the broadest possible access to distributed information. These ontologies meet the requirements and others producing a shared ontology across the participating data sets. In this approach, ontologies are dynamically generated based on a user request that is evaluated against the latest ontological information from each local data source. Participating agents can each understand and provide information about all the data concepts that are shared across the participating systems, which significantly increases the capability of current ontologies. The implementation of this approach produced significant financial benefit, and will see broad deployment in the near future.

10. References


[6] Dr H. Lu, “Ontology based Agent Services Description and Matchmaking on the World Wide Web”, School of Information Technology[HREF1], Deakin University [HREF2], 221 Burwood Highway, Burwood, VIC 3125, AUSTRALIA.


