# Ontologies for Agent-Based Information Retrieval and Sequence Mining

Subrata Das, Kurt Shuster, Curt Wu
Charles River Analytics, Inc.
625 Mount Auburn St.
Cambridge, MA 02138
+1 617 491 3474
{sdas, kshuster, cwu}@cra.com

## ABSTRACT

In this paper, we present two very practical problems in the areas of distributed information retrieval and pattern mining, as well as our proposed solutions via the use of intelligent agents and domain ontologies. The first problem is to retrieve data from heterogeneous distributed data sources with a specific application to distributed Earth Science data archives. Our proposed approach is to develop an engine which acts as an interface agent by presenting users with the appearance of a single, unified, homogenous data source based on a domain ontology of Earth Science terminology. Users can then pose high-level declarative queries against this view. The system then translates each query into a set of sub-queries and spawns mobile agents to retrieve data corresponding to each sub-query. The second problem is to predict significant world events at multiple levels of abstraction by analyzing a collection of events over a period of time in order to generate sequential patterns. We specifically focus on predicting terrorist actions by analyzing terrorist group activities over time. We employ a hierarchical taxonomic organization of contextual event types to obtain higher-level abstractions of observed low-level events. With this approach, significant events can be predicted at multiple levels of abstractions with associated confidences. Although we have addressed these two problems by building prototypes in two different domains, their combination offers a powerful agent-based tool that can assist scientists and analysts by automatically retrieving and mining data collected from multiple distributed data sources. Thus with the use of relevant domain ontologies, the problems of data retrieval and pattern discovery can be combined and automated in a single, elegant system.

## Keywords

Agent, Ontology, Taxonomy, Distributed Information Retrieval, Sequence Mining.

## 1 INTRODUCTION

The exponential growth of the Internet in recent years has given the analysts (e.g. counterterrorism analysts) and scientists (e.g. space and environmental scientists) an opportunity to access large amounts of open-source and classified data that are routinely collected and stored on a continuous basis by many large corporations and government agencies. Some important uses of such data includes predicting future terrorist activities, discovering new space phenomena, and predicting weather patterns and global warming. However, the proprietary nature of these data sources often requires that the data be stored in a number of independent repositories distributed over a network. Because of the large volumes of data stored and the large number of distinct data archives in which the data is located, scientists and analysts often face a daunting task when searching for specific data or series of interrelated data. Moreover, each of these data archives is responsible for a particular domain and autonomously maintains its data in its own distinct format. Consequently, users have to learn the format or metadata information of individual data sources. Thus, we see a need for a tool that would automatically identify and retrieve data from distributed sources based on high-level user queries.

A large amount of research has been directed toward the problem of querying and integrating heterogeneous data from distributed sources. Simplified methods for querying such data sources, which may include traditional databases, knowledge bases, programs, Web pages, and data files, can broadly be categorized into the following two approaches (Widom, 1996): 1) a lazy or on-demand approach, where information is extracted from the sources only when the queries are posed; and 2) an eager or in-advance approach, where relevant information is extracted in advance in anticipation to queries and stored in a central repository. It is simply not practical to create another data repository from several data sources that are already huge and maintained autonomously. Thus adopting an on-demand approach for distributed heterogeneous databases seems quite appropriate, though such an approach to data retrieval requires an infrastructure for retrieving data from distributed data sources based on the query requests. Mobile agent based data retrieval offers several advantages including remote computation, robust to network connection interruption, and autonomy. Such an agent is an autonomous agent with behavior, state, and location.

But irrespective of the approach adopted for integrating heterogeneous distributed data sources, it is necessary to provide users with a single, unified, homogenous interface through which users can then pose high-level declarative queries to retrieve data from distributed data sources. This helps users to avoid the time-consuming process of learning individual data sources. One effective approach to building a unified interface to heterogeneous distributed data sources is via the use of a unified domain ontology. An ontology in a particular domain is a description of the concepts and relationships that can exist in the domain (Sowa, 2000). One of the primary purposes of constructing an ontology is to provide a standard, unambiguous representation of a particular domain of knowledge (Arens et al, 1993). Ontologies have been built and used successfully in constructing multi-contextual knowledge bases, including

common-sense knowledge bases like Cyc (Lenat, 1995), as well as enterprise knowledge (Uschold et al., 1998) and environmental science ontology EDCS (Birkel, 1999). Various ontology representation schemes and acquisition tools are now available, such as XML, Protégé (Noy et. al, 2000), and KIF (Genesereth, 1991).

However, there are several issues that must be addressed during the process of building an ontology for a particular domain:

- Ontological Structure
  - The type of ontology must be chosen based on the given task, with several options available, such as frame-based ontologies, task-based ontologies, and others (Fensel, 2001).
  - Many standardized language choices (e.g. KIF, OKBC)
  - It is often impractical to independently create entire ontologies due to the large size of the domain of interest; therefore several 3rd-party ontologies may need to be integrated.
- Ontology maintenance/evolution
  - Domain may be very specific to a particular field (e.g. oceanic zonation terminology in (Frank and Kemp, 2001)), requiring expert assistance for generation.
  - Ontologies that may change over time must be adaptable.
- Upper-level Ontologies
  - If diverse ontologies must be integrated then semantic discrepancies need to be rectified. This may require a high-level upper ontology (e.g. Cyc upper ontology (Lenat, 1995), SUMO (Niles and Pease, 2001)).
- Populating
  - Much work needs to be done to manually map individual data sources to a global ontology – potentially requiring partial automation of the task.

Additionally, our use of mobile agents for distributed information retrieval raises additional issues regarding their effective operation within an ontological framework:

- Mobile agents and ontologies
  - As agents hop from sites to sites, it is sometimes necessary that each agent carry the entire domain ontology and the translation mechanism for each site it is likely to visit. This approach makes an agent bulkier and therefore slower movements within the network.
  - Mapping from an individual database schema to global ontology is not trivial; programmatic mapping may be required at data source (e.g. converting Farenheit to Celsius). Mobile agents will thus have to carry with them all relevant mapping and translating code.

We are currently addressing the above-mentioned issues within our two ongoing projects: 1) information retrieval from distributed Earth Science data sources (Das, Shuster, and Wu, 2002), funded by NASA; and 2) sequence mining for terrorist threat prediction (Das and Ruda, 2002), funded by DARPA. Our initial focus is to build ontologies in two domains, environmental science and asymmetric threat prediction, including their acquisition via Protégé and subsequent representation in a machine readable XML format. Our approach to the use of ontologies is generic, in the sense that for a particular domain, metadata information from individual sources will be translated to a uniform representation with the use of a single ontology of the domain concerned. Users will pose a query with the ontology in mind and the system will automatically decompose queries into subqueries that are understood by individual data sources.

The rest of the paper is organized as follows. The following section briefly describes the two projects and our approach. Section 3 describes our use of ontologies in these projects, specifically the organization of ontologies in hierarchical taxonomies. Section 4 describes our use of Protégé for acquiring ontologies and their machine readable representations in XML. Finally, Section 5 briefly describes our plan to combine the two approaches into an integrated information retrieval and sequence mining system.

# 2 THE PROBLEMS

This section introduces the two problems that we are currently dealing with and our approach especially with the use of domain ontologies. For more details on theses projects, readers are recommended to read (Das, Shuster, and Wu, 2002; Das and Ruda, 2002).

## 2.1 Information Retrieval from Distributed Earth Science Data Archives (ACQUIRE)

NASA's Earth Science Division continuously collects and stores vast amounts of environmental data for use by a large and diverse community of research scientists, engineers, and analysts. This data comes from a wide variety of sources, including orbiting satellites, weather stations, research aircraft, and others. Various Distributed Active Archive Centers (DAACs) around the globe collect and maintain this data on behalf of NASA; each of these DAACs is responsible for a particular domain and maintains its data in its own distinct format. Researchers who require data stored in these archives often spend a great deal of time locating and integrating the specific data they require. The process would be much simpler and faster if there existed a single, homogenous data repository or the appearance (from the user's point of view) of such a single repository. In this case, the user would not need to 'find' the location of any data, since all of it would appear to be located in the same place. Furthermore, the user could construct his exact query in the form of a suitable database query language, such as SQL.

We have developed (Das, Shuster, and Wu, 2002) an Agent-based Complex QUerying and Information Retrieval Engine (ACQUIRE) for heterogeneous and distributed data sources, and subsequently tested the system on simulated Earth Science data repositories. ACQUIRE implements the following three stages:

- Accepts a query from a user and decomposes it appropriately into a set of sub-queries using site and domain models of the distributed data stores
- Intelligently creates an optimized plan for retrieving answers to these sub-queries over a network and spawns a set of intelligent mobile agents to delegate these tasks

- Appropriately merges the answers returned by the mobile agents and then returns them to the user

Our on-demand approach to data retrieval requires an infrastructure for retrieving data from distributed data sources based on the query requests that are generated from the ACQUIRE front-end. We have used a *mobile agent* approach (Kotz and Gray, 1999), where such an agent is defined as a named object which contains code, persistent state, data, and a set of attributes such as movement history and authentication. A mobile agent can transport itself from one data server host to another as needed for accomplishing its tasks such as searching for relevant data. Such an approach provides distributed querying at sites where the relevant data is available instead of shipping large volumes of data across the network. Unlike remote procedure calls, ongoing interactions do not require ongoing communication in a mobile agent approach. An agent can perform actions with a certain degree of autonomy, such as finding alternate routes in the event of a network failure. Another feature of a mobile agent approach is their ability to carry arbitrary computations to the data storage site. This allows for greater flexibility when retrieving and processing remote data, as relevant data-processing code can be customized to the particulars of a given query. Numerous applications of mobile agents exist, including remote database access, on-line shopping, and communicating with travelers. Some of the commercial-off-the-shelf (COTS) software packages for mobile agents are: IBM's Aglets, Object Space's Voyager, and Mitubishi Electric ITA's Concordia. For our effort we have explored several possible COTS packages for implementing the mobile agents, and we eventually selected the Grasshopper system from IKV Corporation (www.grasshopper.de).

### 2.2 Sequence Mining for Significant Terrorist Action Prediction (TACTICS)

The growing digitization of asymmetric warfare and the exponential growth of the Internet in recent years has given the counterterrorism analyst an opportunity to access large amounts of open-source data. One effective use of such data is for generating past terrorist activity patterns to predict future terrorist activities. However, the manual extraction of hidden patterns within an unorganized large volume of open-source data is nearly an insurmountable task. What is required is an automated technique that will be able to automatically detect useful patterns within gathered data from open sources. We have developed (Das and Ruda, 2002) one such technique where the goal is to make accurate predictions of future events based on extracted patterns from past history and thereby supporting reliable behavior prediction and threat assessment for counterterrorism.

Our recent DARPA-sponsored effort under the TACTICS program has so far been restricted to terrorist activity by a particular terrorist group (the name and other specifics relating to the actual group being studied are not disclosed for reasons of personal security) and its activities during a particular time frame. The past history of the terrorist group activities during the period is represented as a sequence of events. These events include both significant events such as actual terrorist attacks, as well as non-attack events (e.g. leaders visit abroad). In order to represent all the possible events involving terrorist group activities, an event taxonomy has been created that organizes the events into a hierarchical structure. The event taxonomy is applied when events are extracted, and the hierarchical form of the taxonomy is especially useful when only scant information is available about an event. The taxonomy can also be used to generate temporal rules at various levels of abstraction.

The events that are collected from open source and organized hierarchically are then used by machine learning (ML) algorithms to recognize temporal patterns of behavior and to discover behavioral rules. These rules are used to predict future activities based on current data/events. Initial results are promising, indicating that terrorist attacks can actually be predicted with hit rate of 88% (i.e., only 12% of attacks were not predicted) and a false-alarm rate of 37%.

## 3 ONTOLOGIES AND TAXONOMIES

An ontology is an abstract model of a particular field of knowledge. An ontology describes concepts, attributes of concepts, and the relationship between concepts. For example, the taxonomy of species in biology is a type of ontology which classifies all known biological organisms by Kingdom, Phylum, Class, Order, Family, Genus, and Species. The system is hierarchical in nature, such that any organism in the hierarchy posses all of the attributes of the higher-level classification units to which it belongs. For example, Phylum Chordata consists of all animals that have a notochord. Classes Mammalia and Reptilia both belong to this phylum, and thus they both share the common attribute of possessing a notochord. An instance is a concrete instantiation of a particular class within the ontology. So whereas "African Elephant", "Grey Wolf", and "Saber-toothed Tiger" represent different species within the ontology of organisms, "Dumbo", "Spot", and "Fluffy" are specific instances of those species. A knowledge base is a data structure which contains both an ontology and specific instances.

One of the primary purposes of constructing an ontology is to provide a standard, unambiguous representation of a particular domain of knowledge. This facilitates communication between domain experts in a given field. If a biologist discovers a new species, she can specify its kingdom, phylum, etcetera, and other biologists will understand without ambiguity the attributes of the new species, since they all share the same vocabulary. The following two subsections describe the use of ontologies and taxonomies in two of our ongoing projects ACQUIRE and TACTICS.

### 3.1 Ontology of Earth Science Data in ACQUIRE

In ACQUIRE, the domain of discourse is Earth Science data, and thus we require an ontology of Earth Science terms, including standard definitions of space, time, weather, etcetera. This ontology serves as a common reference linking the diverse and nonuniform naming schemes used in the various data sets stored in NASA's DAAC system. For example, data from two different DAACs sets may contain temperature data for different regions of the earth. One data source may store the temperature in a column labeled "temp", while the other uses "temperature". To resolve this issue (known as the polymony and synonymy problem), ACQUIRE's common earth science ontology will contain a TEMPERATURE class that unambiguously denotes all

temperature measurements. All data sources accessible to ACQUIRE will require a mapping between the data set's idiosyncratic naming convention and ACQUIRE's universal ontology. Thus both the "temp" data and the "temperature" data can both be accessed with a single query for TEMPERATURE. Note that there are two distinct mapping steps in the process. The first mapping is done off-line when the data source is added to ACQUIRE's list of available repositories. A system administrator must perform this one-time mapping, known as data modeling, for each data source when the data source is added. The second mapping is the dynamic data acquisition performed by ACQUIRE during actual data retrieval. The software automatically performs this operation whenever a data source is accessed, thus providing the 'transparency' of the system's data retrieval functionality.

A second reason for employing an ontological approach to data retrieval is that it allows for a much greater flexibility in query structure. For example, a researcher may wish to know the total precipitation over a given region and time period. Specific NASA archives may store various types of precipitation (e.g. one that stores snowfall over a given region, another that stores rainfall). If a user wants to know the total precipitation, he would have to query both snowfall and rainfall data sources independently, and then combine the results. With an ontological approach, he can simply specify "precipitation" in his query, and the system would automatically recognize snowfall and rainfall as subclasses of precipitation. The system will then return all data sets that store rainfall, snowfall, and any other type of precipitation. Alternatively, he can simply specify "snowfall" in his query, and the system would then only retrieve "snowfall" data sets.

As Earth Science data is the primary type of information stored at NASA's DAACs, it is necessary to create an ontology of Earth Science terms, data types, etc. Because Earth Science data typically involves measurements of a particular region at a particular time, the ontology must include two primary measurement types: those of spatial and temporal values (Bishr and Kuhn, 2000). Although most information stored in the DAAC system is geospatial in nature, much of the data contain extremely domain-specific terminology. For example, an ontology of oceanic zonation terms (Frank and Kemp, 2001) is shown below in Figure 1 and Figure 2.

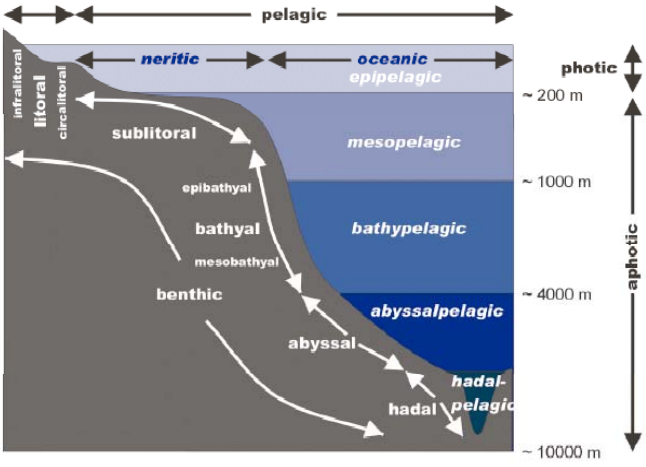


**Figure 1: Oceanic Zonation (Frank and Kemp, 2001)**

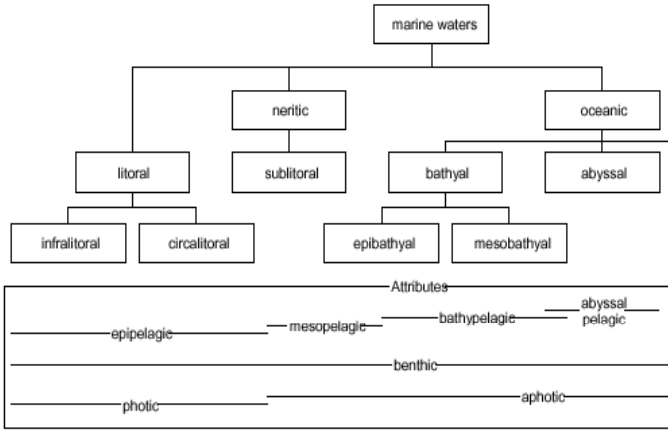


**Figure 2: A Marine Ontology (Frank and Kemp, 2001)**

Due to this high level of specificity, it is essential that third-party ontologies created by domain experts be easily integrated with ACQUIRE's high-level upper ontology. Integrating diverse ontologies will be crucial for realizing NASA's goal of a distributed, virtually-centralized, and semantically-rich database system. Until recently, however, a major problem with integrating diverse ontologies has been the lack of a high-level upper ontology to serve as a foundation for more domain-specific ones. Typically, domain-specific ontologies either define their own high-level concepts or leave them out entirely. These high-level semantic differences between diverse domains have restricted the integration of ontologies from vastly different fields. The Suggested Upper Merged Ontology (Niles and Pease, 2001) is an IEEE effort to create a standard upper ontology which will allow semantic integration of diverse domain ontologies through shared high-level concepts. ACQUIRE will utilize SUMO as a foundation for the automatic integration of domain-specific ontologies for large, heterogeneous data sources.

In ACQUIRE, a "query" is an abstract data type that encapsulates both a request for data any data-processing code to be applied to that data. A query is generally constructed from a higher-level "interface query" which depends on the particular user interface being employed. For example, ACQUIRE could employ an SQL interface in which the user enters a query as a standard SQL string. This string would then be translated to ACQUIRE's internal query structure before being decomposed into individual subqueries to be retrieved by mobile agents. Alternatively, the interface may be a natural language system that takes English sentences as input and translates that input into ACQUIRE's internal query representation. This way, ACQUIRE can accommodate any interface so long as it translates the user's request into ACQUIRE's internal query data structure. The details of this data structure are beyond the scope of this paper, but in general the structure is much like that of a parsed SQL query, with additional fields corresponding to any data processing code.

Once the query is requested by the ACQUIRE interface, it must be decomposed into a series of subqueries corresponding to the actual physical location of the data and the particulars of the data schema used. This is done in three primary stages:

First, ACQUIRE breaks the query into retrieval units based on the physical location of the data types requested. So, if the query requires data of type "atmospheric-ozone" and "polar-ice-

thickness", the system queries its catalog of data sites that contain data of this type, and creates a retrieval agent for each one. In this example, "atmospheric-ozone" and "polar-ice-level" were previously defined in the ontology of Earth Science terminology, and any data sources containing information of this type was previously cataloged by an administrator.

The next step is to optimize the query. Suppose the query was for all polar ice thickness measures taken when atmospheric ozone levels were above a certain threshold. The system would prioritize the retrieval by first retrieving all atmospheric ozone levels and then direct the polar ice retrieval agents to only retrieve polar ice from those regions and times.

The final step is to map each agent's ontology-based data type against the data schema of the data site at which it is stored. For this process, a wrapper is created which maps the particulars of the data site schema to the ontology-based description. So, if a data site stores polar ice thickness in a relational database table called "ICE" and a column called "THICKNESS", the wrapper would consist an appropriate SQL query that selects THICKNESS data from table ICE. The wrapper also contains any data-processing code required. So if the thickness data is stored in feet but the user wants it in meters, then translation code will be sent along with the agent to perform the translation at the site. Additionally, if the query requested only the mean values, then code to perform this (or any other) statistical operation will also be included.

Once the query is decomposed and the retrieval agents generated, the system spawns the mobile agents and waits for the results to return, at which time it merges the results and presents them to the user via the user interface. Notice that, in some cases, some agents may not leave until other ones have returned with required intermediate data, as described above.

It should be noted that in the current incarnation of ACQUIRE, all data accessible by the system must be manually modeled and mapped against the global ontology. Clearly, any attempt to integrate large numbers of data sites will require a substantial manual data modeling effort. In addition, any changes to the data sites already mapped must be remapped against the data site catalog. One potential solution to this problem would be to send agents to unmapped data sites along with the entire domain ontology and code for automated data site mapping. Work on the Cyc project (Lenat, 1995) has been done in the area of automated database understanding, and such an approach could be used with our mobile agents to determine site contents. This approach still has many inherent problems to overcome, however, such as the large size of the agents required to transmit both the ontology and data-analysis code.

Another problem to address is that of unit type translation at the data source. For example, one site may store temperature data in Celsius while another used Fahrenheit units; data translation code must therefore be sent along with the mobile agents if remote computation is to be done at the distributed data sites. Although the mapping between Celsius and Fahrenheit is trivial, many such mappings are not. For example, a data site may contain concentrations of a certain pollutant, say $S0_2$, in a data table, while another stores such information in an image with various concentrations represented by different colors. Queries requiring a combination of both data sources would therefore require a much more complex data translation algorithm; it is hard to imagine a system in which this type of translation would not require customized processing code for each data site representation.

## 3.2 Ontology and Taxonomy in TACTICS

In TACTICS, the domain of discourse is terrorist threat prediction, and thus we have defined an ontology of terrorist activity terms, including standard definitions of attack, threat, propaganda, etcetera. The past history of the terrorist activities during the period considered is represented as a sequence of events. These events include both significant events such as actual terrorist attacks, as well as non-attack events (e.g. leaders visit abroad). The procedure for collecting the events using the developed ontology is currently semi-automated. Newspaper articles and other sources are searched for connections to the group under consideration, and matching articles are stored in a database. Trained analysts then scrutinize these articles for events, and any events are represented according to the event type taxonomy (discussed below) and stored in the database as well. The extracted events are then used by a sequence learning engine to generate meaningful temporal rules.

We have developed a taxonomy for contextual event types for a terrorist group. Contextual events form the top node of the hierarchy, and represent incidents that occur in regions of interest and can be related to the group being studied. The taxonomy for contextual event types is shown in Figure 3. The set of all contextual event types have been categorized into direct events, regular occurrences, and indirect events.

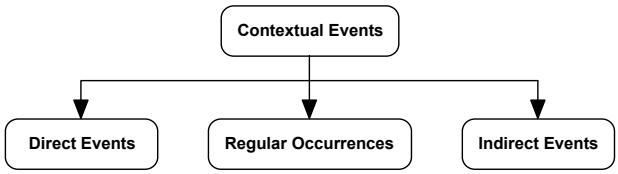


**Figure 3: Taxonomy for Contextual Events**

Direct events are incidents that can be directly related to the group. Figure 4 shows that the set of all direct events have been categorized into action/activity by group, action/activity against group, action/activity against population, action/activity in favor of group, and peripheral events. Of these five sub-categories, we focus on the action/activity by group category that includes events resulting from actions directly executed by group members.

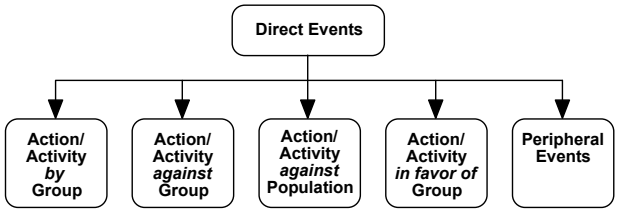


**Figure 4: Taxonomy for Direct Events**

A portion of the structure of the action/activity by group category is shown in Figure 5. Group members carry out various types of activities including political actions, the execution of missions, threats of missions (often related to planning), and changes in their goals and modus operandi. Each of these types is

further sub-classified until it is refined to a level of classification that cannot be specified any further. These atomic actions or activities by the group at the leaf nodes of a hierarchy are directly observable and reported in the open source literature. For example, a bombing that results in the outcome of death is a specific observable event with a clear classification.

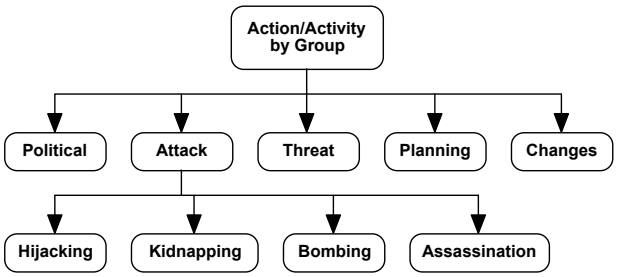


**Figure 5: Partial Taxonomy for Actions/Activities by Group**

On the other hand, the executed mission/attack type is at a higher level of abstraction and does not specify which type of mission is being undertaken. For example, given three hijacking and two kidnapping actions, one could abstract the knowledge that five missions were executed without specifying the nature of the missions. This kind of organization helps to generate predictions of terrorist actions at various levels of abstraction and confidence. For example, consider the following three rules where the number after each rule represents its confidence and where 100% signifies absolute confidence:

IF Militants Captured and Jailed THEN Hijacking (30%)

IF Militants Captured and Jailed THEN Kidnapping (20%)

IF Militants Captured and Jailed THEN Hijacking & kidnapping (10%)

The above three rules can be combined by adding the confidences of the first two rules and subtracting the confidence of the third rule, which is the intersection of the sets, to generate a rule with higher level of abstraction:

IF Militants Captured and Jailed THEN Attack (40%)

If the event Militants Captured and Jailed occurs then both terrorist actions Hijacking and Kidnapping would be predicted at different confidence levels, but the terrorist action Attack, which is more abstract than Hijacking and Kidnapping, would be predicted at a higher level of confidence. This kind of prediction is useful when it is very important just to be aware of a terrorist threat irrespective of its type.

# 4 ONTOLOGY ENCODING

This section describes our use of Protégé for acquiring ontologies and their representation in a machine readable XML format.

## 4.1 Protégé-2000

A Knowledge Representation System (KRS) is a tool for constructing knowledge bases. A KRS contains a set of protocols that define the allowable structure of a particular ontology. Loom (isi.edu/isd/LOOM), Protégé-2000 (protege.stanford.edu), and Ontolingua (ksl.stanford.edu/software/ontolingua) are examples of well-known knowledge representation systems. These three systems all conform to the Open Knowledge-Base Connectivity (OKBC) protocol, which specifies a set of minimum requirements for interoperability between knowledge bases (http://www.ai.sri.com/~okbc/). For ACQUIRE, we are using the Protégé-2000 KRS developed by Stanford Medical Informatics (http://protege.stanford.edu/index.shtml).

Protégé is both a Knowledge Representation System and a graphical development tool. It is available free of charge, free from licensing conditions, for all commercial and educational purposes. It is actively updated and supported by its creators at SMI, and has a large and diverse user community. Protégé is being used by ACQUIRE for three purposes: as a representation language for an ontology of earth science data; for modeling data sites and data sets against the ontology; and for querying the data sets. These three functional features will each be described in detail below.

As a knowledge representation language, Protégé offers a number of beneficial features. The primary one is its compatibility with the OKBC protocol, which allows it to easily integrate partial ontologies that are themselves OKBC compliant. Protégé also supports multiple inheritances, which allows class membership in more than one parent class. Finally, ontologies constructed with Protégé can be easily modified and extended without the need for major refactoring of the ontology's existing structure. This is important because the ontology is likely to be 'dynamic', in that it will change over time as the development team gains more experience with the salient concepts of ontology construction. In the longer term, this is important because even well-constructed ontologies are likely to change over time as scientific information changes (for example, the taxonomy of species often changes as scientists discover new species or when they learn that known species were previously misclassified).

Data Modeling in ACQUIRE involves: 1) Ontology generation: defining the semantic types of information available from all sources; 2) domain modeling: the description of the actual objects and tables in a data source; and 3) site modeling: the description of the site where a data source resides. We have started exploring the use of Protégé-2000 for all three aspects of data modeling. An example of ontology generation using Protégé is shown in Figure 6 below.

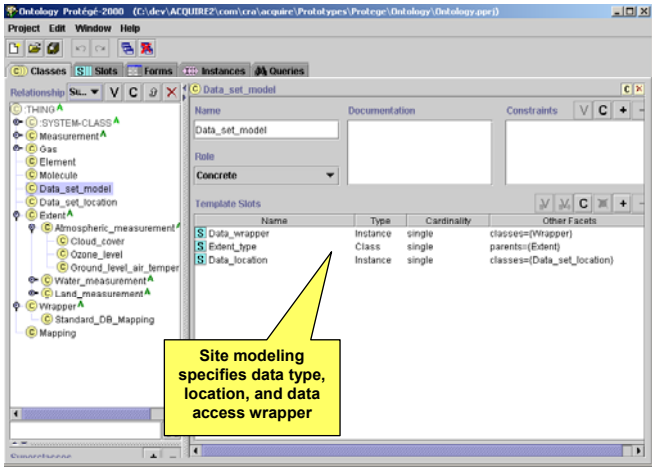


**Figure 6: Site and Domain Model Ontology**

Once an ontology is created in Protégé, it can be populated with instance data. An instance is a concrete instantiation of a particular class within the ontology (see Figure 7 below). This process of populating the ontology specifically maps the physical location (site modeling) and access information (domain modeling) to the abstract data representation language specified by the ontology. The site model tells the system where to find a data set within the network, while the domain model defines the actual names of tables and columns within that data set. Figure 8 shows a portion of the text file output corresponding to this ontology.

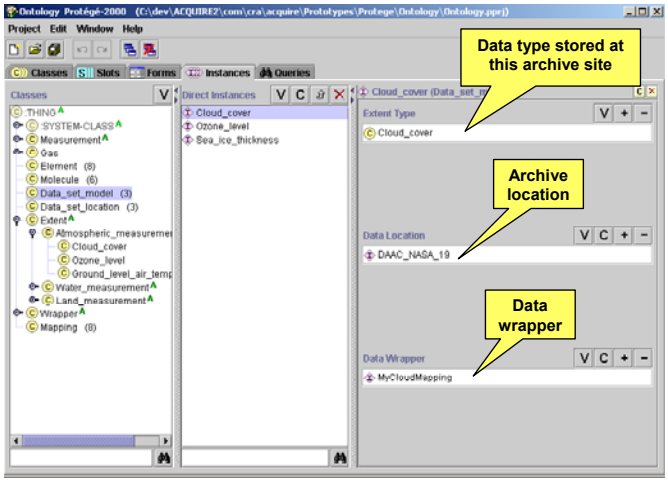


**Figure 7: Site Model Instance Data**

```
(defclass Data_set_model
            (is-a USER)
            (role concrete)
            (single-slot Data_wrapper
                         (type INSTANCE)
;+                       (allowed-classes Wrapper)
;+                       (cardinality 0 1)
                         (create-accessor read-write))
            (single-slot Extent_type
                         (type SYMBOL)
;+                       (allowed-parents Extent)
;+                       (cardinality 0 1)
                         (create-accessor read-write))
            (single-slot Data_location
                         (type INSTANCE)
;+                       (allowed-classes Data_set_locati
;+                       (cardinality 0 1)
                         (create-accessor read-write)))

(defclass Data_set_location
            (is-a USER)
            (role concrete)
            (single-slot name_
                         (type STRING)
;+                       (cardinality 0 1)
                         (create-accessor read-write))
            (single-slot Repository_URL
                         (type STRING)
;+                       (cardinality 1 1)
                         (create-accessor read-write)))
```

**Figure 8: Ontology Encoding in Protégé**

## 4.2 XML

In TACTICS, both the event-type taxonomy and the location taxonomy are stored in XML-based text files. XML provides an excellent storage format because it is a good compromise between both human and machine readability, and editing the appropriate file easily extends a taxonomy. The structure of the XML file uses only a total of three tags and three attributes are used. The nesting of the elements reflect the hierarchy of the taxonomy. The basic element used is the <Node>, which has a required "name" attribute, specifying the name of the node. The other attributes that may be assigned to the <Node> element are "key" and "ref". The "key" attribute is used to give a node a unique reference name, for those cases where the name attribute is not unique. The "ref" attribute is used when branches in the hierarchy are joined, and specifies a unique <Node> name, or a ref value. The other two elements are <Alternate>, which only uses the "name" attribute, and <Comment> which places an arbitrary comment between the element begin and end tags. The <Alternate> element is used to specify an alternate spelling for a <Node> name. This is especially useful for alternate spellings of place names, dealing with different languages, contractions, and even misspellings. A sample of the XML used to describe the event-type taxonomy is shown in Figure 9 below. The sample demonstrates the use of the tags and attributes discussed above.

```
<Node name="Shooting">
  <Node name="Leader" key="Leader2"/>
  <Node name="Member" key="Member2">
    <Alternate name="Members"/>
    <Alternate name="member"/>
  </Node>
  <Node name="Civilian" key="Civilian2">
    <Alternate name="Civilians"/>
    <Alternate name="Civilian Shooting"/>
    <Alternate name="Shooting Civilian"/>
    <Alternate name="Shooting-Civilian"/>
  </Node>
</Node>
<Node name="Imprisonment">
  <Alternate name="Imprisonement"/>
  <Alternate name="Imprisonemnt"/>
  <Alternate name="Imrisonment"/>
  <Node ref="Leader2">
    <Alternate name="Leader Imprisonment"/>
    <Alternate name="leader Imprisonment"/>
    <Alternate name="Leader Imprisonement"/>
  </Node>
  <Node ref="Member2">
    <Alternate name="Member Imprisonment"/>
    <Alternate name="Imprisonment Member"/>
    <Alternate name="Member Imprisonemnt"/>
  </Node>
  <Node ref="Civilian2">
    <Alternate name="Civilian Imprisonment"/>
  </Node>
</Node>
```

**Figure 9: XML Fragment from the Event Type Taxonomy**

## 5 COMBINING OUR APPROACHES

We have seen how ontologies can be used for sequence mining of terrorist threats and for the retrieval of heterogeneous and distributed data. Although we have not yet done so, we foresee much potential for a system that combines these two approaches into a single, comprehensive system. Such a system could potentially automate the task of sequence discovery in large bodies of scientific data, such as NASA's massive Earth Science data archives. Because of the tremendous volume of such data, sequence mining and other knowledge discovery methods traditionally require large, time-consuming data transfers. With a mobile agent approach, the data can be analyzed for sequences at the storage site, thus allowing a much larger corpus of data to be analyzed.

One of the drawbacks of the TACTICS system is that data must be fed in manually from news sources such as newspaper articles and TV reports. An automated data retrieval system that collects news items from a database could substantially facilitate data acquisition. This would, of course, require a suitable ontology of news article 'topics', along with a significant amount of manual work dedicated to classifying news archives against this ontology. Research in the filed of automatic text understanding and classification would certainly be relevant here.

## 6 CONCLUSIONS

In this paper, we have presented two very practical problems in the areas of distributed information retrieval and pattern mining, and raised and addressed several issues in relation to our use of intelligent agents and domain ontologies as proposed solutions to the problems. We have described our use of Protégé for constructing ontologies and subsequent representation in a machine readable format. Our future plan is to continue addressing the issues that are raised in Section 1, including the ones related to the use of existing domain ontologies such as Cyc and EDCS. We will then address the task of combining the process of information retrieval with pattern discovery by using a single domain ontology to accomplish both tasks concurrently.

## 7 REFERENCES


[1] Arens, Y., Chee, C. Y., Hsu, C-N., In, H., and Knoblock, C. A.. (1993). Retrieving and integrating data from multiple information sources. *International Journal on Intelligent and Cooperative Information Systems*, Vol. 2, pp. 127-158.

[2] Birkel, P. (1999) "SEDRIS Data Coding Standard", In *Proceedings of the Spring Simulation Interoperability Workshop*, March 1999, 99S-SIW-011.

[3] Bishr, Y. and Kuhn, W. (2000) Ontology-Based Modelling of Geospatial Information. In *Proceedings of the 3rd AGILE Conference on Geographic Information Science*, Helsinki/Espoo, May 25-27.

[4] Das, S., Shuster, K., and Wu. C. "Agent-based Complex Querying and Information Retrieval Engine", to appear in the *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-agent Systems (AAMAS 2002)*, Bologna, Italy, July 2002.

[5] Das, S. and Ruda, H. "Predicting Significant Events via Sequence Learning", to be presented at the *ECAI Workshop on Knowledge Discovery from Temporal and Spatial Data*, Lyon, France, July 2002.

[6] Fensel, D. (2001) "*Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*". Springer-Verlag.

[7] Foley, P. Mamaghani, F. & Birkel, P. The Synthetic Environment Data Representation and Interchange Specification (SEDRIS) development project (http://www.sedris.org/pr11trpl.htm).

[8] Frank, R. and Kemp. Z. (2001) Ontologies for Knowledge Discovery in Environmental Information Systems. In Raffacto A and Renso C, editors, *International Conference Logic programming ICLP'01 Workshop Proceedings CRGD: Complex Reasoning on Geographical Data*, December 2001.

[9] Genesereth, M. R. (1991). "Knowledge Interchange Format". In *Proceedings of the Second International Conference on the Principles of Knowledge Representation and Reasoning (KR-91)*, Kaufman, pp 238-249.

[10] Kotz, D. and Gray, R. (1999). Mobile Agents and the Future of the Internet. *ACM Operating Systems Review*, August 1999, pp. 7-13.

[11] Lenat, D. B. "Cyc: A Large-Scale Investment in Knowledge Infrastructure." *Communications of the ACM* 38, no. 11 (November 1995).

[12] Niles, I. and Pease, A. (2001). Towards a Standard Upper Ontology. In C. Welty and B. Smith (Eds.) *Formal Ontology in Information Systems: Collected Papers from the Second International Conference*. New York: ACM Press, pp. 2-9.

[13] Noy, N. F., Fergerson, R. W., and Musen, M. A. (2000). The knowledge model of Protege-2000: Combining interoperability and flexibility. *2nd International Conference on Knowledge Engineering and Knowledge Management (EKAW'2000)*, Juan-les-Pins, France, 2000.

[14] Sowa, J. (2000). *"Knowledge Representation"* Brooks/Cole.

[15] Uschold, M., King, M., Moralee, S., and Zorgios, Y. (1998) The Enterprise Ontology *The Knowledge Engineering Review*, Vol. 13, Special Issue on Putting Ontologies to Use (eds. Mike Uschold and Austin Tate). (Also available from Artificial Intelligence Application Institute (AIAI), University of Edinburgh, Scotland, as AIAI-TR-195).

[16] Widom, J. (1996). "Integrating Heterogeneous Databases: Lazy or Eager?", *ACM Computing Surveys*, Vol. 2.