

SEMANTIC INTEROPERABILITY IN DIGITAL LIBRARIES
USING INTER-ONTOLOGICAL RELATIONSHIPS

by

SRIRAM LAKSHMINARAYAN

B.E., University of Madras, India, 1998

A Thesis Submitted to the Graduate Faculty
Of The University of Georgia in Partial Fulfillment
of the
Requirements for the Degree
MASTER OF SCIENCE

ATHENS, GEORGIA

2000

© 2000

Sriram Lakshminarayan

All Rights Reserved

SEMANTIC INTEROPERABILITY IN DIGITAL LIBRARIES
USING INTER-ONTOLOGICAL RELATIONSHIPS

by

SRIRAM LAKSHMINARAYAN

Approved:

Major Professor

Date

Approved:

Graduate Dean

Date

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Amit Sheth, without whom this thesis could not have been completed. I would also like to thank Dr. Dan Everett and Dr. Naveen Ashish for serving on my committee. It has been a rewarding experience working on the ADEPT¹ project in the LSDIS lab. Last but not the least, I would like to thank Tarcisio Lima for extending the much needed help during the course of completing the thesis and of course, my colleagues for their cooperation.

¹This work was funded in part by the National Science Foundation (NSF) under Grant No. IS IRI-9411330. The ADEPT team consists of the University of California (Santa Barbara and Los Angeles), the University of Georgia and the San Diego Supercomputer Center.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS.....	iv
LIST OF FIGURES.....	vii
CHAPTER	
1 INTRODUCTION.....	1
2 STATE OF THE ART.....	4
2.1 THE PROBLEM: SEMANTIC HETEROGENEITY.....	4
2.2 SOLVING THE PROBLEM.....	4
2.3 GENERAL CONCEPTS.....	5
2.4 COMPARISON OF RELATED ARCHITECTURES AND WORK....	6
3 THEORETICAL ASPECTS.....	14
3.1 DESCRIPTION OF ONTOLOGIES.....	14
3.2 ONTOLOGIES AND SEMANTIC INTEROPERABILITY.....	20
3.3 SIGNIFICANCE OF THE “AFFECTS” RELATIONSHIP.....	21
3.4 DESIGN OF THE “AFFECTS” RELATIONSHIP.....	22
3.5 USE OF FUZZY MAPPING FUNCTIONS AND OPERATORS.....	27
3.6 RELATE: A RELATIONSHIP INFORMATION STORE.....	30
3.7 EXAMPLE RELATIONS.....	31
4 SYSTEM IMPLEMENTATION.....	34

4.1		
4.2	SYSTEM ARCHITECTURE.....	34
4.3	THE ONTOLOGY AGENT.....	37
4.4	THE RELATIONSHIP MANAGER.....	39
4.5	SYSTEM DEMONSTRATION.....	43
5	CONCLUSION.....	46
5.1	CONTRIBUTIONS.....	46
5.2	FUTURE WORK.....	47
	BIBLIOGRAPHY.....	48
	APPENDICES	
A.	ONTOLOGY DTD.....	
	52
B.	RELATIONSHIP DTD.....	
	53

LIST OF FIGURES

3.1 NATURAL DISASTER ONTOLOGY.....	15
3.2 VOLCANO ONTOLOGY.....	16
3.3 EARTHQUAKE ONTOLOGY.....	17
3.4 TSUNAMI ONTOLOGY.....	17
3.5 ENVIRONMENT ONTOLOGY.....	18
3.6 COUNTRY ONTOLOGY.....	19
3.7 HOW DO VOLCANOES AFFECT THE ENVIRONMENT?.....	23
3.8 HOW DOES HEAT AFFECT ANIMALS?.....	26
3.9 HOW DO VOLCANOES AFFECT THE ENVIRONMENT?.....	27
3.10 HOW DOES AN AUTO TRANSMISSION AFFECT PERFORMANCE?.....	32
3.11 HOW DOES PLASMODIUM AFFECT HUMAN HEALTH?.....	32
3.12 HOW DO ELECTRICAL MACHINES AFFECT THE RADIO?.....	33
4.1 SYSTEM ARCHITECTURE.....	35
4.2 GRAPHICAL ONTOLOGY BUILDER.....	38
4.3 RELATIONSHIP BUILDER.....	42
4.4 ISCAPE SELECTION.....	43
4.5 RESULTS.....	44

4.6 SIMULATION OPERATION..... 45

4.7 SIMULATION RESULTS.....45

ABSTRACT: SEMANTIC INTEROPERABILITY IN DIGITAL LIBRARIES USING INTER-ONTOLOGICAL RELATIONSHIPS

Information gathered from diverse sources tends to be semantically heterogeneous. Correlation of this kind of information is the topic of this thesis. We try to achieve semantic interoperability by the use of ontologies and inter-ontological relationships. We develop a framework for the definition of ontologies and a schema for generic definition of relationships. We also develop an information store that handles various information requests using the relationships and utilizes mapping functions for spatial and temporal matching. Of special interest is the “affects” relationship, which plays a powerful role in describing interactions among the elements, especially in the real world. We develop a prototype system based on the Digital Earth paradigm and demonstrate its use in learning environments.

CHAPTER 1

INTRODUCTION

The Web as we know is a vast source of information. It grows at an astounding rate too. Estimates are that the current number of public web pages is around 800 million. The sources of information are equally different or heterogeneous, in technical terms. There can be several forms of heterogeneity – syntactic, structural and semantic [KAS00]. We use the Web, in large part as an information source and efficiently retrieving relevant information in a coherent form is an area of active research.

Finding relevant information on the Web may not be an easy task. Various tools exist that help us in our quest. Web directories like Yahoo! [YAH00] classify information into categories that reflect human thinking. Search engines do a reasonably good job of retrieving information, depending on what one searches for and the search technology. For example, Google [GGL00] ranks Web pages on the basis on the number of links to it. Others search based on pure keyword match.

It is a difficult task to construct systems which are capable of answering intelligently, since the system would need to “understand” the terms involved and the relationships between them. It should also be noted that the answer to an information request might require accesses to multiple information sources. Additionally, the information is usually semi-structured and is present in a wide variety of formats.

What we require for this kind of a system - and what most existing commercial systems lack - is the ability to semantically relate information. Several research projects have sought to reduce this gap (Section 2.4). We would soon see the realization of a dream known as the “Semantic Web” – a web of data that can be processed by machines directly or indirectly [BER99]. There are a couple of hurdles to cross before we can achieve this, though – the biggest being semantic heterogeneity.

We need to work with and utilize information from heterogeneous data sources – something we call semantic inter-operability. Inter-operability is achieved by the effective use of metadata, and shared knowledge sources called ontologies (Section 2.3). A powerful construct is that of inter-ontological relationships, that helps us relate terms and entities between ontologies. This is the core of semantic inter-operability, something we will delve in depth in this thesis. Of special interest is the *Affects* relationship, which is a powerful mechanism for describing relationships in natural domains like geography. It helps us define schematically, questions like “How does A affect B?” and answer them using the system.

We implement all these ideas on a test-bed application called the Alexandria Digital Earth Prototype (ADEPT) for the geographic domain based on a geo-referenced digital library. This system is based on a distributed architecture that solves the problem of semantic heterogeneity. We place special emphasis on accessing distributed heterogeneous geo-spatial information and the application of the system in learning environments [PAL00].

The rest of this thesis is outlined as follows. Chapter 2 describes the current technology in solving the heterogeneity problem along with related ideas and concepts. Chapter 3 describes the theoretical concepts behind the working of the system including ontologies, relationships, and their detailed design. Chapter 4 describes the actual implementation of the ideas on the ADEPT system. Finally, Chapter 5 concludes this work and describes future improvements that can be made.

CHAPTER 2

STATE OF THE ART

In this chapter, we discuss general concepts of ontologies, relationships, information brokering and mediation. We also compare the ADEPT system with various related systems that perform information integration.

2.1 THE PROBLEM: SEMANTIC HETEROGENEITY

Semantic interoperability is a major challenge in information gathering and mediation. Semantic heterogeneity represents the mismatch in meanings when independently designed systems are integrated. There are various kinds of conflicts that cause semantic inconsistencies like name, structure, attributes, granularity of values, etc. The fundamental axiom of mathematics $X=X$ is invalid in the real world of natural expression (which is prevalent on the web), where individuals may express themselves in the most effective manner [WIE97]. Thus, we need semantic interoperability to go beyond simple understanding of terms and comprehend the deeper meaning.

2.2 SOLVING THE PROBLEM

What is semantic interoperability? The ability of a user to access consistently and coherently, similar classes of digital objects and services, distributed across

heterogeneous repositories, with mediating software compensating for site-by-site variations [CHE99].

So, how do we achieve this? There are several ways to solve the heterogeneity problem. We follow the approach of using a hybrid architecture that supports both Information Brokering and Mediation (Section 4.1) along with the use of ontologies (to bridge the gap between heterogeneous data sources), metadata (to represent the data in a compact and homogeneous format), and inter-ontological relationships (for interoperability between domains).

2.3 GENERAL CONCEPTS

What is an ontology? According to Gruber [GRU00], an ontology is a specification of a conceptualization. A conceptualization is an abstract, simplified view of the world that we wish to represent for some purpose. An ontology is a description of the concepts and relationships that can exist between them. We design ontologies for the purpose of enabling knowledge sharing and reuse. In that context, an ontology is a specification used for making ontological commitments. An ontological commitment is an agreement to use a vocabulary in a way that is consistent with respect to the theory specified by an ontology.

Let us now define a relation. Given a set X , a relation is some property that may or may not hold between one member of X and another [SET00]. For instance if attention is restricted to sets of numbers then the operator " $<$ " for "is less than" is a valid relation, as would be " $=$ " for equality. If R stands for any general relation, then we write " $x R y$ " to show that x relates to y under R , where x and y are members of the sets X and Y

respectively. Relations are not only things that arise with numbers. In a tree or ontology, the relations “is parent of” or “is child of” are relevant. We shall see later on that these ontological relationships play a vital role in semantic interoperability.

The ADEPT prototype developed at the University of Georgia (ADEPT_{UGA}) allows us to pose information requests or high level queries in the form of what are called Information Landscapes (Iscales). These allow complex questions like “How do volcanoes affect the environment?” to be answered by the access to information from multiple heterogeneous sources.

2.4 COMPARISON OF RELATED ARCHITECTURES, PROJECTS AND IDEAS

Several research systems out there currently address similar issues. The major theme in all these systems, of course, is semantic interoperability and information mediation. We shall see how they work and perform in comparison to our system:

2.4.1 INFORMATION MEDIATION SYSTEMS

Mediator based architectures consist of wrappers for encapsulating heterogeneous information sources to provide uniform interface to the rest of the world, and mediators to provide a broad variety of value added services [KAS00]. Wiederhold [WIE92] proposed the concept of a mediator as a way of formulating semantic information necessary to integrate heterogeneous sources and to make sense out of a collection of potentially incomplete and inconsistent data and inherently incompatible programs.

2.4.2 INFORMATION BROKERING ARCHITECTURES

Information brokering architectures guide creation and management (brokering) of systems and solutions to serve the information and business needs of a variety of information stakeholders [KAS00]. The information requested by users can come from a variety of information sources. This requires arbitration between the user and the information providers in the form of explicating the semantics and (re) interpretation of the information/query [KAS94].

2.4.3 RELATED IDEAS ON SEMANTICS AND RELATIONSHIPS

There is a lot of research being done in the area of relations in diverse domains like language, philosophy and information systems. [SEM00] describes how semantics is important in language and how improper use can lead to ambiguity. [RLR00] describes a relational repository for keyword semantic proximity, similar to the RELATE manager developed (Section 3.6). The Ontolingua [ONT00] system developed at Stanford allows web-based editing of ontologies and taxonomies. The OBSERVER system [MKS96] has an Inter-Ontologies Relationships Manager, similar to the RELATE store developed in ADEPT, which focuses on synonym/hypernym/hyponym relationships. The InfoSleuth [BKV97] system employs an Ontology Agent similar in function to the one in ADEPT.

2.4.4 THE TSIMMIS PROJECT

The Stanford-IBM Manager of Multiple Information Sources [CGH94] is a system for integrating information. It offers a data model and a common query language that are designed to support the combining of information from many different sources. It also

offers tools for generating automatically the components that are needed to build systems for integrating information. The TSIMMIS architecture uses the mediator architecture [WIE92] and the principle components are as described below:

- A lightweight object model called OEM (Object Exchange Model) serves to convey information among components and is so called because it does not require strong typing of its objects;
- A common query language called LOREL (Lightweight Object Repository Language) is used to link components and query substructures in OEM objects. This is similar to the language used by the metadata broker to represent metadata query expressions;
- Translators or wrappers allow LOREL queries to be converted into source-specific queries. The information sources are not necessarily databases, and it is an important goal of the project to cope with radically different information formats in a uniform way. They provide the functionality given by the translators and the wrappers in the metadata system in the abstract metadata-based architecture;
- Mediators use LOREL both to ask and answer queries. Some example mediators used in TSIMMIS are the union mediator that takes queries and passes them unchanged to two or more other sources, and the join mediator that creates a view of two other sources.

The emphasis in the TSIMMIS system is that of automatic generation of translation and mediators for accessing and combining information in heterogeneous data sources. When these are indeed generated, they constitute a system which performs brokering at the level of information content, where a high level data model (OEM) and language (LOREL)

may be used to query information in data repositories storing structured or unstructured data. The problem of using different vocabularies to construct LOREL expressions describing similar information is not tackled.

2.4.5 THE SIMS PROJECT

The goal of the SIMS project [ACH93] is to provide intelligent access to heterogeneous distributed information sources, while insulating human users and application programs from the need to be aware of the location of the sources, their query languages and organization, etc. A model of the application domain is created using a knowledge representation system to establish a fixed vocabulary describing objects in the domain, their attributes, and relationships among them. For each information source a model is constructed that indicates the data-modeling used, query language, network location, size estimates, etc. and describes the contents of its fields in relation to the domain model. Queries to SIMS are written in the high-level uniform language of the domain model, a language independent of the specifics of the information sources. Queries need not contain information describing which sources are relevant, where they are located or how the information obtained from different sources should be combined or manipulated. SIMS uses a planning system to determine how to retrieve and integrate the data necessary to answer a query. The planner selects the appropriate information sources, orders the sub-queries to the appropriate information sources, etc. to determine an optimized plan for distributed execution of the query and initiates execution. The different modules are as follows:

- Modeling: SIMS provides a uniform way to describe the application domain and the information sources to the system, so that data in them is accessible;
- Information source selection: Given a query, SIMS determines the information sources containing the data relevant for answering the query. For queries which appear to have no matching information source, it determines if any knowledge encoded in the domain model can be used to determine relevant information sources;
- Query plan optimization: SIMS constructs a plan for retrieval of the information requested by the query. The plan involves steps such as sending a specific query to some information source and joining the results from different information sources;
- Execution: The fourth component executes the optimized plan. To support execution, SIMS makes use of wrappers that mediate between it and the information sources themselves.

The system performs brokering primarily at the level of information content and partially at the level of vocabulary. The user can query heterogeneous data repositories by using high level expressions in the SIMS knowledge representation language. Since the focus is within the application domain, it lacks inter-domain adaptability. However, it does support intra-domain adaptability via the query reformulation operations.

2.4.6 THE OBSERVER SYSTEM

The OBSERVER system [MKS96] is an instantiation of the abstract metadata-based architecture for information brokering. Brokering is performed at the level of information content and at the level of vocabulary. The components are as described below:

- Query processor: The query processor represents the vocabulary broker identified in the abstract metadata-based architecture. It adds extra functionality by enabling vocabulary brokering across information domains. An expression constructed using concepts from a chosen user ontology is taken as input. A query may either be partially or fully translated into the component ontologies;
- Inter-ontology relationships manager: terminological relationships between terms in different ontologies are represented in a declarative manner in an independent repository. These relationships enable different vocabulary brokering functions;
- Metadata system: The metadata system is responsible for enabling brokering at the level of information content. The main components of the metadata system are:
 - Ontology server: It accesses the ontologies stored in the metadata repository and retrieves the data underlying the ontology corresponding to a given expression;
 - Metadata repository: The metadata repository is responsible for the definitions of the terms in the metadata and also for the mappings between the terms in an ontology and the underlying data structures in the local data repositories.

The system performs brokering at the metadata level, although the focus is on vocabulary brokering utilizing relationships across terms in different ontologies.

2.4.7 THE INFOSLEUTH SYSTEM

The InfoSleuth system [BKV97] is an agent-based system for information gathering and analysis. The functionality of the various components are as described:

- User agent: The user agent is the user's gateway to the network of InfoSleuth agents. It handles requests from the user, routing those requests to appropriate server agents and passing responses back to the user;
- Broker agent: The broker agent acts as a matchmaker which pairs requests from user agents to other resource agents that can fulfill that request. As agents come online, they can advertise their information using semantic metadata expressions to the broker agent. The advertised metadata descriptions are stored in a metadata repository which is tapped into by the broker agent;
- Ontology agent: The ontology server is responsible for managing the creation, update and querying of ontologies belonging to the multiple domains;
- Multi-resource query agent: The multi-resource query agent is responsible for the execution of ontology-based queries. It decomposes the queries into sub-queries based on its knowledge of appropriate resource agents that can satisfy the query, and forwards the decomposed queries to the selected agents;
- Resource agents: The resource agent acts as an intelligent front end interface to the data repositories, accepting high-level ontology-based queries from the network and translating them into the local repository query language. Results are then translated back into the terms of the ontology and returned to the requesting agent.

Infosleuth uses domain specific ontologies as the basis for brokering across multiple information sources containing information that can be described by a common domain ontology.

2.4.8 PREVIOUS WORK IN ADEPT

Pervious work in this area including [BER98] and [RAY99] have addressed some problems of semantic interoperability. New additions in our system include an enhanced agent based architecture, support for inter-ontological relationships, tools for building ontologies and relationships, operations like simulation and an user interface incorporating a learning model.

CHAPTER 3

THEORETICAL ASPECTS: ONTOLOGIES AND RELATIONSHIPS

In this chapter, we discuss the development of various ontologies in the system and discuss their significance. We then describe the *Affects* relationship, its design and use in the system using a generic schema developed. We then show how we can use these relationships in an effective way using a relationship store, along with the use mapping functions to achieve geo-spatial and temporal interoperability.

3.1 DESCRIPTION OF ONTOLOGIES AND SCHEMAS

We have currently developed ontologies for volcanoes, earthquakes, tsunamis and environment in the natural disasters domain, among others, which are used in our system.

They are as described below:

3.1.1 NATURAL DISASTER ONTOLOGY

What is a natural disaster? It is a natural calamity resulting in widespread destruction and loss – something that man is powerless to prevent. There are several common disasters that we know of. Examples are volcanoes, earthquakes, floods, etc. Figure 3.1 shows a representation the first-level natural disaster ontology.

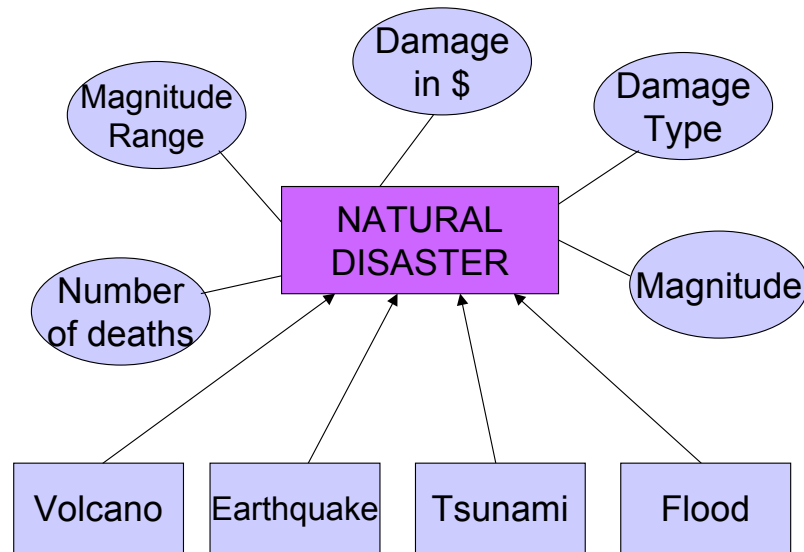


Fig. 3.1 Natural disaster ontology

3.1.2 VOLCANO ONTOLOGY

A volcano is any vent in the crust of the Earth or other planet or satellite, from which issue molten rock, pyroclastic debris, and steam [BRI00]. Volcanoes are closely associated with tectonic activity. Most of them occur on either the overriding or the diverging margins of the enormous lithospheric plates that make up the Earth's surface. The volcanoes of Japan provide an excellent example of the former, while those of the Mid-Atlantic Ridge the latter. Intra-plate volcanoes such as those of the Hawaiian chain provide important evidence as to the direction and rate of plate motion.

Volcanoes affect humankind in many ways. Their destructiveness is awesome, but the risk involved can be reduced by assessing volcanic hazards and forecasting eruptions. Volcanism provides fertile soils, valuable mineral deposits, and geothermal energy. Over geologic time volcanoes recycle the Earth's hydrosphere and atmosphere, and explosive eruptions can affect climate. The partial volcano ontology is shown in figure 3.2.

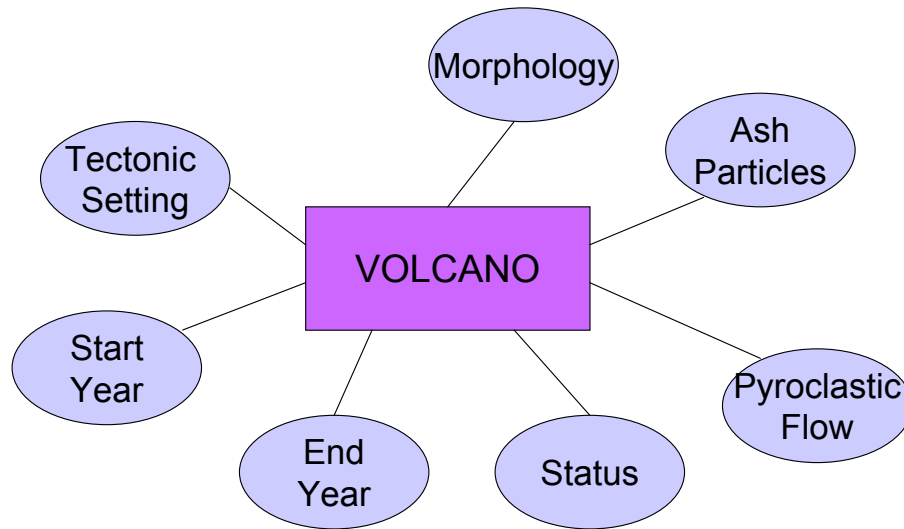


Fig. 3.2 Volcano ontology

3.1.3 EARTHQUAKE ONTOLOGY

“An earthquake is any sudden disturbance within the Earth manifested at the surface by a shaking of the ground” [BRI00]. This shaking, which accounts for the destructiveness of an earthquake, is caused by the passage of elastic waves through the Earth's rocks. These seismic waves are produced when some form of stored energy, such as elastic strain, chemical energy, or gravitational energy, is released suddenly.

Few natural phenomena can wreak as much havoc as earthquakes. Over the centuries they have been responsible for millions of deaths and an incalculable amount of damage to property. While earthquakes have inspired dread and superstitious awe since ancient times, little was understood about them until the emergence of seismology at the beginning of the 20th century. A partial earthquake ontology is shown in figure 3.3.

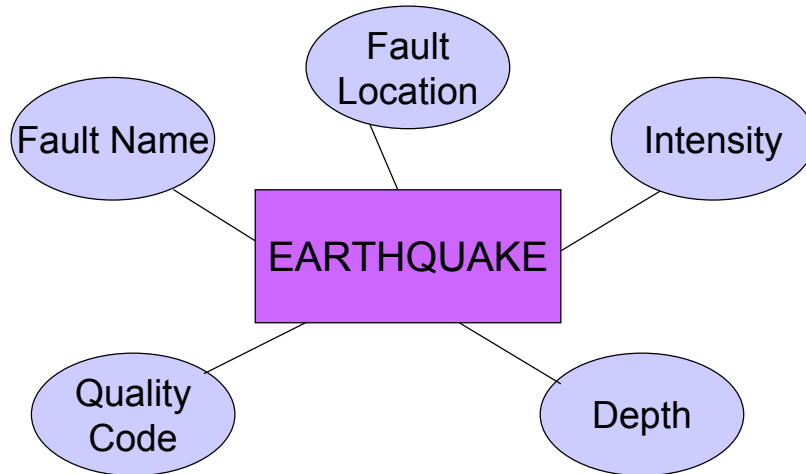


Fig. 3.3 Earthquake ontology

3.1.4 TSUNAMI ONTOLOGY

A Tsunami - also called as a seismic sea wave or tidal wave – is usually caused by a submarine earthquake occurring less than 30 miles beneath the seafloor, with a magnitude greater than 6.5 on the Richter scale [BRI00]. Underwater or coastal landslides or volcanic eruptions also may cause a tsunami. Tsunamis can savagely attack coastlines, causing devastating property damage and loss of life. A partial tsunami ontology is shown in figure 3.4.

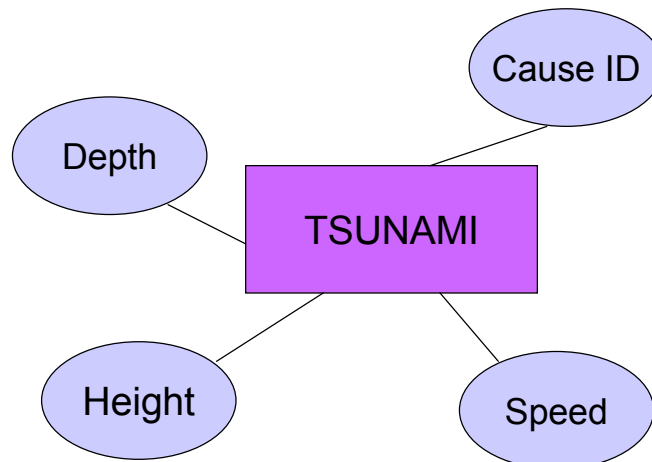


Fig. 3.4 Tsunami ontology

3.1.5 ENVIRONMENT ONTOLOGY

Environment is a very vast domain and it does not make sense to construct it fully. A more viable approach is to model only the sub-domain of interest. A general representative set of themes would encompass energy, natural disasters, pollution, hydrological cycle and human living among others [RAY00]. Our current interest is in the area of natural disasters, and hence the specific sub-domain has been modeled as shown in figure 3.5.

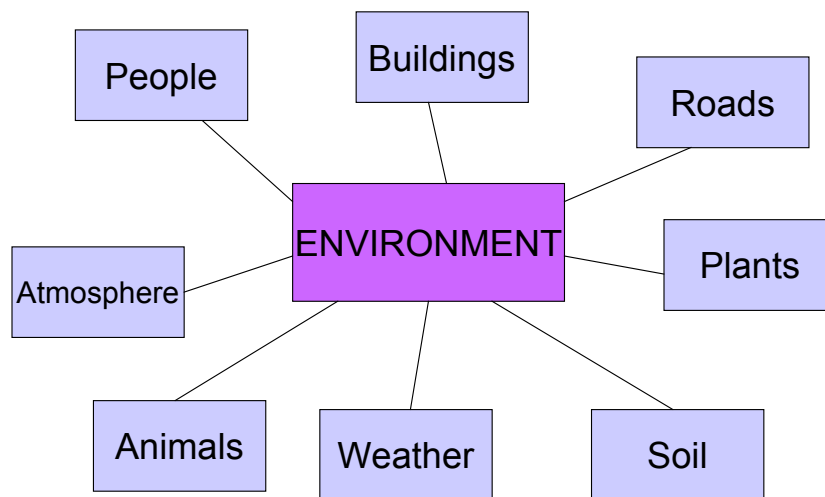


Fig. 3.5 Environment ontology

3.1.6 COUNTRY ONTOLOGY

A country is represented by its people, government, economy, natural resources and other entities. A complete schema would be rather huge. We here try to represent the most relevant attributes needed for our Iscares. Shown in figure 3.6 is the partial country ontology.

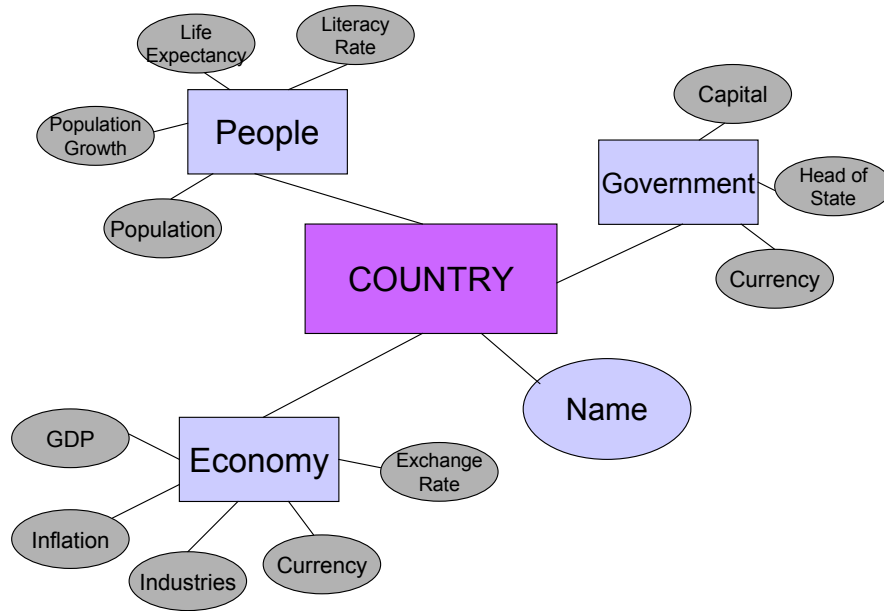


Fig. 3.6 Country ontology

3.1.8 ONTOLOGY SCHEMA

In our effort of realizing the “Semantic Web”, we use ontologies as a method of inter-operating between heterogeneous data. The ontologies are all defined using a common schema, which helps in regularizing the different domains. We have developed an XML schema for defining ontologies in a systematic and homogeneous way. The main identifying properties of an ontology are name, keywords associated, description, attributes along with their descriptions, and the parent ontology. Any ontology can be described in XML using these properties as shown below:

```
<?xml version="1.0"?>
<!DOCTYPE Ontology SYSTEM "ontology.dtd">
<Ontology>
  <Name>Volcano</Name>

  <Keywords>
    <Keyword>Sesmic</Keyword>
    <Keyword>Lava</Keyword>
    <Keyword>Natural Disaster</Keyword>
  </Keywords>

  <Description>This is a volcano ontology</Description>
```

```

<Attributes>
  <Attribute
    Name="Location"
    Alias="Location"
    Type="Char"
    Desc="Location of the volcano">

  </Attribute>
  <Attribute
    Name="Name"
    Alias="Name"
    Type="Char"
    Desc="Name of the volcano">
  </Attribute>
  <Attribute
    Name="Height"
    Alias="Height"
    Type="Integer"
    Desc="Height of the volcano in meters">
  </Attribute>
</Attributes>

<Parent>Natural Disaster</Parent>

<ResourceAgents>
  <Agent Name="VolcanoResourceAgent">
    <Attrib>Name</Attrib>
    <Attrib>Location</Attrib>
    <Attrib>Type</Attrib>
  </Agent>

  <Agent Name="NaturalDisasterResourceAgent">
    <Attrib>Damage</Attrib>
    <Attrib>Deaths</Attrib>
  </Agent>
</ResourceAgents>
</Ontology>

```

The last tag, <ResourceAgents> is not part of the ontology per se, but is used by our system for describing which resource agent handles the particular attributes.

3.2 ONTOLOGIES AND SEMANTIC INTEROPERABILITY

The current problem with information obtained from web and other sources, is that they are readable only by humans. The ability to exchange information without misunderstanding in an automated fashion is the key to interoperability.

Bridging the gap among multiple information sources is an active area of research. Ontologies play a very important role in achieving this. They capture domain knowledge in a generic way and provide a commonly agreed understanding of the domain. They are used as a basis for structuring and simplifying the process of constructing domain-specific problem solving tools.

Ontologies consist of terms and a precise definition of those terms that can be leveraged to build meaningful higher level knowledge. Ontologies explicate the contents, essential properties, and relationships between terms in a knowledge base [MIT00]. Bridging the semantic gap between heterogeneous sources to answer end user queries is a prerequisite and key challenge to support global information systems. The basis for this bridge is found in an ontology. Ontologies relate to knowledge sources just as dictionaries relate to literary works. Each application or component typically has individual ontologies in addition to the shared ones, but uses the shared ontologies to exchange data between applications.

3.3 SIGNIFICANCE OF “AFFECTS” RELATIONSHIP

There are various relationships that have been studied by others in the past, for example the “is a”, “is part of” and “like” relationships. If we consider similarity-based relations [KAS00], we have three major types:

- Synonyms: When two different ontologies have the same semantics, they have a synonym relationship with each other. “like” is an example;
- Hyponyms: When a term in one ontology is semantically more specialized than another term in another ontology, they have a hyponym relationship;
- Hypernyms: When a term in one ontology is semantically more general than another term in another ontology, they have a hypernym relationship.

However, most of these relations are hierarchical or similarity based. These are not powerful enough for our task of semantic interoperability between domains like Geography and other which we deal with in our day to day lives, where we have natural “affects” relations between the ontologies. We introduce the general “affects” relationship to cover this gap.

3.4 DESIGN OF THE “AFFECTS” RELATIONSHIP

We will deal with a general statement: How does A *affect* B? A statement like this is pretty open. We start by looking what it means. As real world entities, A due to its properties can affect several other entities in different ways. In a similar way, B can be affected by several entities, of which A may be one. We now try to define how A affects B:

A, in its entirety or by a set of its components, induces some changes or properties on a set of components of B.

Let us see how we can develop a formal model to this end, so that it can be used in a computable environment. An example would be appropriate here to put things in perspective: As shown in figure 3.7, how do volcanoes affect the environment? It should

be noted that this is an illustrative example only and does not accurately reflect the real world phenomenon.

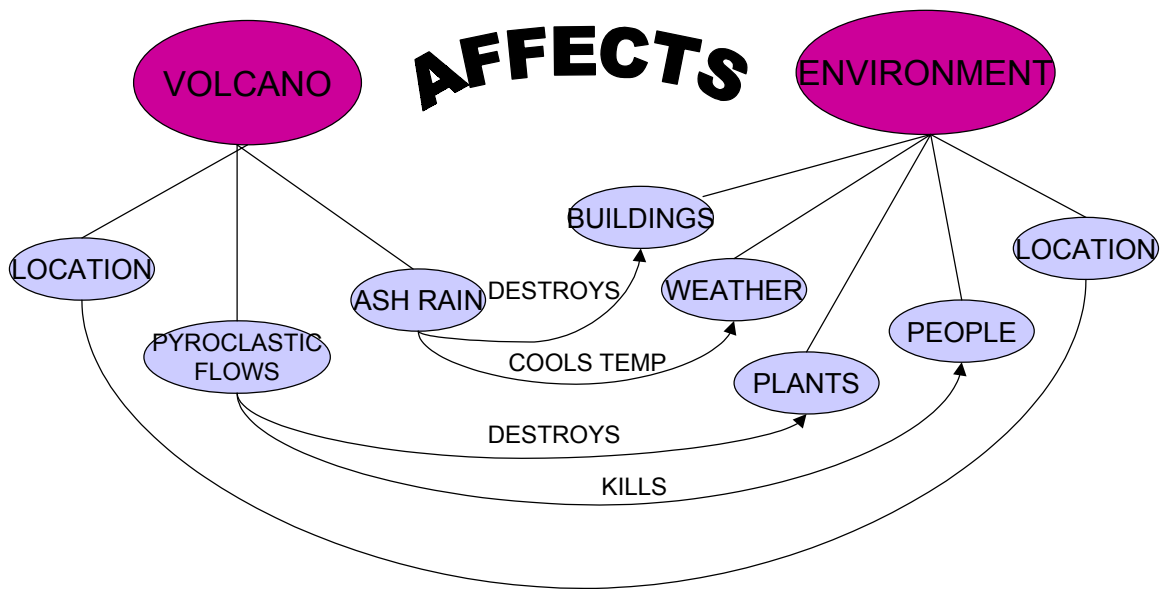


Fig. 3.7 How do volcanoes affect the environment?

Here, the A and B ontologies are ‘Volcano’ and ‘Environment’ respectively. Each of them have sub-components, and it can be seen that the components have an effect on each other. We can immediately see that this relation is composed of sub-relations, which in this case are:

- Pyroclastic flows *kills* People;
- Ash Rain *destroys* Buildings.

According to our earlier definition, a set of components {Pyroclastic Flows, Ash Rain} of A (Volcano) affect a set of components {People, Buildings} of B (Environment). These sub-relations may have some conditions involved with them if they have to be true. Also, these conditions are the only computable entities, using which we can determine whether the given relation holds for the set of ontologies at hand. It should be noted that these

conditions are not strictly necessary to generate a relationship statement. We can have cases where we do not have any computable entities in our relation, in which case, it just becomes a statement describing how the relationship works. In this case, we do have relations and they are:

- Pyroclastic flows *kill* People IF the area of the flows *intersects* the area of people habitat
- Ash Rain *destroys* Buildings IF the volume of ash rain deposited on buildings *is greater than* a threshold value.

Here, Pyroclastic flows and Ash rain are nouns represented as ontologies. Each of them also have a “qualifier”, in this case the words “area” and “volume”. These qualifiers can be represented as enclosing functions for the ontologies and its components. When wrapped with the enclosing functions, they provide for a homogeneous way of comparing or relating the ontological attributes. For example, Area (Volcano.PyroclasticFlows) and Volume (Volcano.AshRain). Of course, we may have more than one set of conditions to be evaluated for the given sub-relation to hold. In general, we can define the relationships as follows:

$$(\exists x \mid x \in \text{ASC}) \text{ and } (\exists y \mid y \in \text{BSC})$$

$$[\text{FN}(x) \text{ operator } \text{FN}(y)]^* \Rightarrow [\text{ASC relation BSC}]$$

$$[\text{ASC relation BSC}]^* \Rightarrow A \text{ affects } B$$

where:

6 A, B are ontologies

ASC is a set containing the sub-components of A

BSC is a set containing the sub-components of B

FN is a an enclosing function belonging to FunctionSet

operator is a set belonging to OperatorSet

relation is a set belonging to RelationSet

FunctionSet, OperatorSet and RelationSet are defined for the specific domain(s) that we are dealing with. FunctionSet typically defines the different enclosing functions that would be used in the relationships. OperatorSet defines the various relational operators that are used. It should be noted that these operators are overloaded – they provide different functionality for different enclosing functions. For example, the = operator would have to equate weight in a way that is different that equating volume. Finally, the RelationSet defines the different sub-relations that form part of the main relationship in the particular domain. For the geography domain and the example above,

FunctionSet = {Area, Location, Time, Volume}

OperatorSet = {<, >, =, <=, >=, INTERSECT}

RelationSet = {Increases, Decreases}

A complete expression for the example above would be modeled as follows:

[Area (Pyroclastic Flows) INTERSECT Area (People Habitat)]

=> [Pyroclastic Flows *kill* People]

[Location (Volcano) = Location (Buildings)] AND [Volume (Ash Rain) > 100]

=> [Ash Rain *destroys* Buildings]

=>

[Volcano] *affects* [Environment]

So far, we have looked at simple relationships. We can also have what are called “Transitive” relationships, which are of the type:

$$A \text{ affects } B \text{ and } B \text{ affects } C \Rightarrow A \text{ affects } C$$

Let ASC be as defined earlier and CSC be defined similarly. Let BSC’ be the set containing a subset of components of B and BSC’’ be another subset. Then a transitive relation would hold only if:

$$ASC \text{ affects } BSC' \text{ and } BSC'' \text{ affects } CSC \text{ and } BSC' \cap BSC'' \neq \{\phi\}$$

This means that transitive relations work only if the intersection of the subset of relations affected by A and the subset of relations that affect C is not null. “How does heat affect animals?” is an example to conceptualize how transitive relations work.

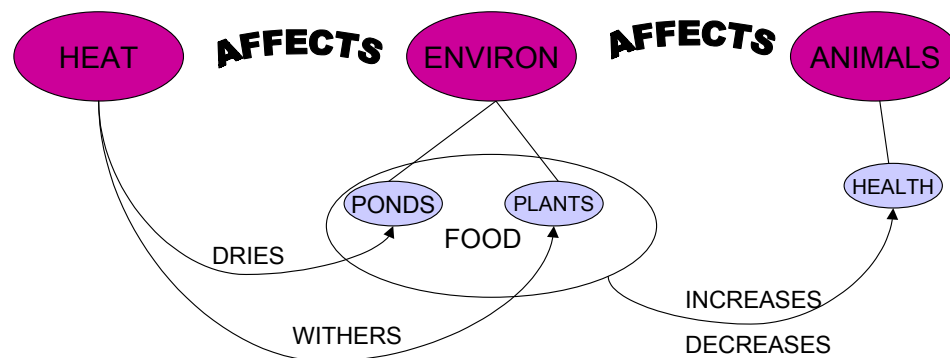


Fig. 3.8 How does heat affect animals?

Here, we have two component relations that make up the whole relation:

- How does heat affect the environment?
- C. Which can be represented as [Heat *dries up* ponds] and [Heat *withers* crops]
 \Rightarrow [Heat *affects* the Environment]
- How does the environment affect animals?

Which can be represented as [*Decrease* (Pond water) *affects* Health (Animals)] and
 [Loss (Plants) *affects* Health (Animals)] => [Environment *affects* Animals]

It should be noted here that both the relations do not have any computable effects at this level of granularity. We could definitely model how heat dries up ponds using complex parameters and further calculate how that affects animal health. Our model is scalable in this aspect and one need only model the relationship to the level desired.

3.5 USE OF FUZZY MAPING FUNCTIONS AND OPERATORS

Let us get back to the detailed constructs of the relationship, especially the conditionals used to qualify the sub-relations. Reconsider this:

$$[FN(x) \textit{ operator} FN(y)]^* \Rightarrow [ASC \textit{ relation} BSC]$$

The construct on the left-hand side is the crux of the relationship computation, and warrants a more detailed inspection. Remember that FN is specific to the domain in consideration and that *operator* can be overloaded. Let us revisit the *Volcano affects Environment* example.

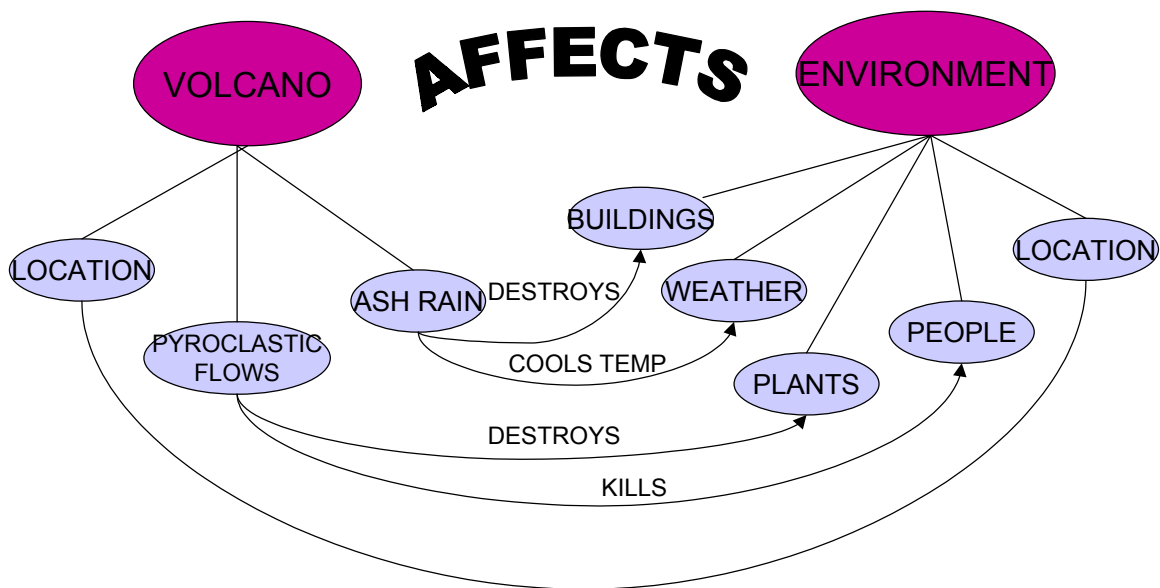


Fig. 3.9 How do volcanoes affect the environment?

Let us first describe the major relationships involved:

- [Area (Pyroclastic Flows) INTERSECT Area (People habitat)] and [Time (Volcano) = Time (Environment)] => [Pyroclastic Flows *kill* People]
- [Area (Ash Rain) INTERSECT Area (Buildings)] and [Volume (Ash Rain) > 100 cubic meters] => [Ash Rain *destroys* Buildings]
- [Location (Volcano) = Location (Environment)] and [Size (Ash Particles) < 2 micron] and [Height (Ash Eruption) > 500 meters] => [Volcanic Eruption *cools* the Environment]

The domain of discourse here is natural disasters and the FunctionSet described is {Area, Location, Time, Size, Magnitude, Height, Depth}. We also use the *operator* INTERSECT, in addition to the regular relational operators defined earlier. Let us take a closer look at some of the interesting operations:

- [Location (Volcano) = Location (Environment)]
Location of the volcano is geo-referenced, i.e., calculated as a point using the latitude and longitude values. Location of a given place is similarly calculated. The “Location” function takes an ontology and returns a point. If we were to compare the two locations using this singular point, we would hardly get any matches. The idea here is to do an approximate match. We know that the effects of a volcanic eruption would not be focussed at a point – rather, they would be felt around an area surrounding the volcano. This essence has to be captured in the functioning of the ‘=’

operator with respect to location. This helps solve geo-referential and/or spatial discrepancies. To compare locations, we may assume that the effects of the volcano are felt around a 10 mile or any other user-defined radius. In that case, we need to check whether the given location point falls within the 10 mile circular area. The '=' operator in this case is overloaded with respect to the "Location" function. The concept of overloading is not new, but fits in nicely in the current context.

- [Time (Volcano) = Time (Environment)]

Here, we compare the time frame in which the volcano erupted and whether it affected the environment at the specified time. For example, if the volcano erupted on a particular day, presumably its effects were felt for a week around its surroundings. Thus, if the specified date for the location in question is a day later, we might suggest that the volcano does indeed have effects on that date. On the other hand, if topic at hand was earthquakes, then the time frame is likely to be in seconds, or at the maximum, in minutes. Thus given a time and date, we can do an approximate match based on the entity at hand. This is also called temporal matching.

- [Area (Pyroclastic Flows) INTERSECT Area (Crops)]

Here, we have to determine if the path of pyroclastic flows from the volcano covers the area of crops. We need to define two areas – one of the pyroclastic flows and the other for crops. This is done by the "Area" function. Then we need to see whether there is an intersection between the two boundaries. The INTERSECT operator does this with respect to area.

- [Size (Ash Particles) < 2 microns]

In this case, the “Size” function gives the numerical size of the given ash particles.

This is a standard comparison using the ‘<’ operator and no overloading is done.

The main theme in using the enclosing functions and overloaded operators is that this scheme provides for inter-operating with entities that are of different granularity and type. This helps achieve inter-operability at the lower levels.

3.6 RELATE: A RELATIONSHIP INFORMATION STORE

Putting all the ideas together, we arrive at the concept of a relationship store. This is where we keep track of:

- The different ontologies created;
- The different relationships created;
- The synonym/hyponym/hypernym relations between the ontologies.

This combination of information items gives us a powerful manipulating scheme and allows us to infer several facts about the data present. Some of the interesting things that can be done are:

- Questions and answers: This is the simplest of queries – that of explaining how one entity relates to B. This is achieved by accessing the required relation and displaying the individual sub-relations that make up the whole;
- General questions: Since we have ready access to all the relationships and the ontological hierarchy, we can pose generalized questions like “How do all natural disasters affect the environment?”. Even though we may not have direct information about natural disasters affecting the environment, the repository contains information about how entities which are natural disasters (those which have natural disasters as the parent) affect the environment. Clubbing this information allows us to give an

AI-like answer such as “I don’t know. But I know that volcanoes, earthquakes and tsunamis affect the environment in the following ways”. This involves recursively splitting the question till we can gather information about the relation at that level. For this, we define two types of ontologies: Root and Leaf. Leaf ontologies are the ones that are most specific and do not need further processing. Root ontologies need to be traversed till their leaves and a sub-query generated for each level;

- Open ended questions: Questions like “What are the factors affecting B?” or “What all does A affect?” can be answered with ease, since these types of questions essentially require multiple matching at either end;
- Automatically inferring transitive relationships: Every time a relationship “A affects B” is created, the repository is cross-checked to see if any other entities affect A or if B affects any entity. Once a match is found, the subset of attributes of the common entity is examined to make sure the intersection is not null (as explained in section 3.4);
- Use of the “like” relationship: Using our synonym store, we can construct queries using the “like” relation. Since we know that two entities are synonyms, they are interchangeable. Thus, any query posed with an ontology not described in our system, but described as a synonym for one of the existing ones, will yield an answer.

3.7 EXAMPLE RELATIONS

We will now explore a couple of relations from diverse domains and show how all of them can be described using our schema.

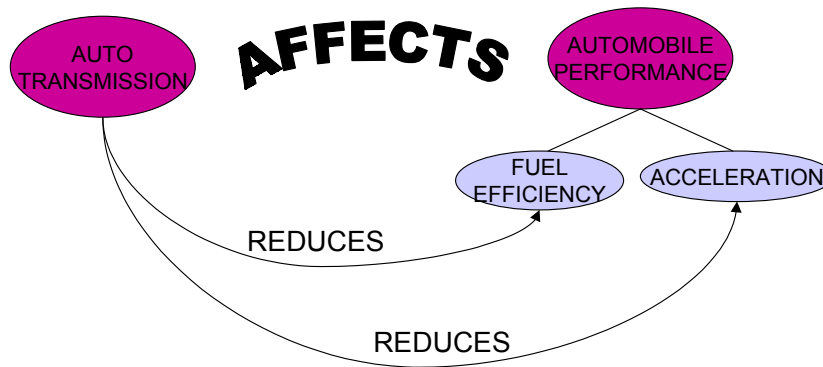


Fig. 3.10 How does an auto transmission affect automobile performance?

[Auto Transmission *reduces* Fuel Efficiency] and [Auto Transmission *reduces* Acceleration] => [Auto Transmission *affects* Automobile Performance]

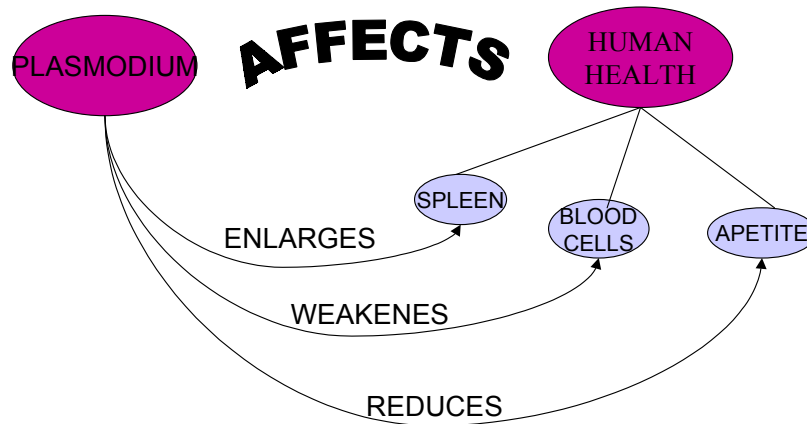


Fig. 3.11 How does Plasmodium affect Human Health?

[Plasmodium *enlarges* Spleen] and [Plasmodium *weakens* Blood Cells] and [Plasmodium *reduces* Appetite] => [Plasmodium *causes* Malaria] => [Plasmodium *affects* Human Health]

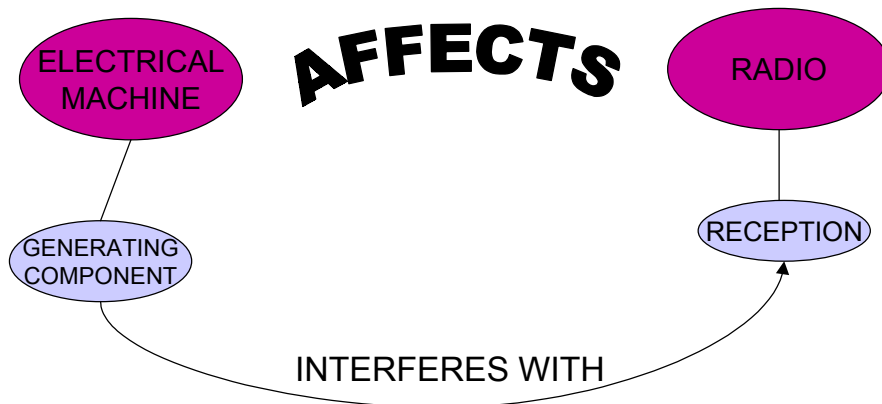


Fig. 3.12 How do Electrical Machines affect the Radio?

[Frequency (Generating Component) > 1000 Hz] => [Generating Component *interferes with* Reception] => [Electrical Machines *affect* the Radio]

We see here that we build up the information base with very specific knowledge. Each relationship described is built upon smaller sub-relationships. Combined, they give the user the information needed. This schema is scalable and can be used to build very complex relationships.

CHAPTER 4

SYSTEM IMPLEMENTATION

The realization of semantic interoperability can be seen in our Alexandria Digital Earth Prototype (ADEPT) system [ADE99]. Of special concern here is accessing geo-spatial information of all types and the application of the system in learning environments [PAL00]. It should be noted that ADEPT is only one of our target applications. All the concepts are generic and domain independent, though.

4.1 SYSTEM ARCHITECTURE

We have developed a distributed multi-agent architecture for the ADEPT system (Figure 4.1). The agents that interact with each other are the User agent, the Ontology agent, the Planning agent, the Correlation agent and the Resource agent. They interact with each other through a common Broker. It should be noted that the different agents may reside on different machines. The different agents are described in detail below:

4.1.1 THE USER AGENT

The User agent is the primary interface between the system and the user. It serves in different capacities depending on the mode, which can be Student or Administrator. In the Administrator mode, it allows the user to build Iscares with a learning model. The

Student mode allows the user to execute Iscapes and learn about the subject through multi-media views and results. For a more detailed study, refer [PAL00].

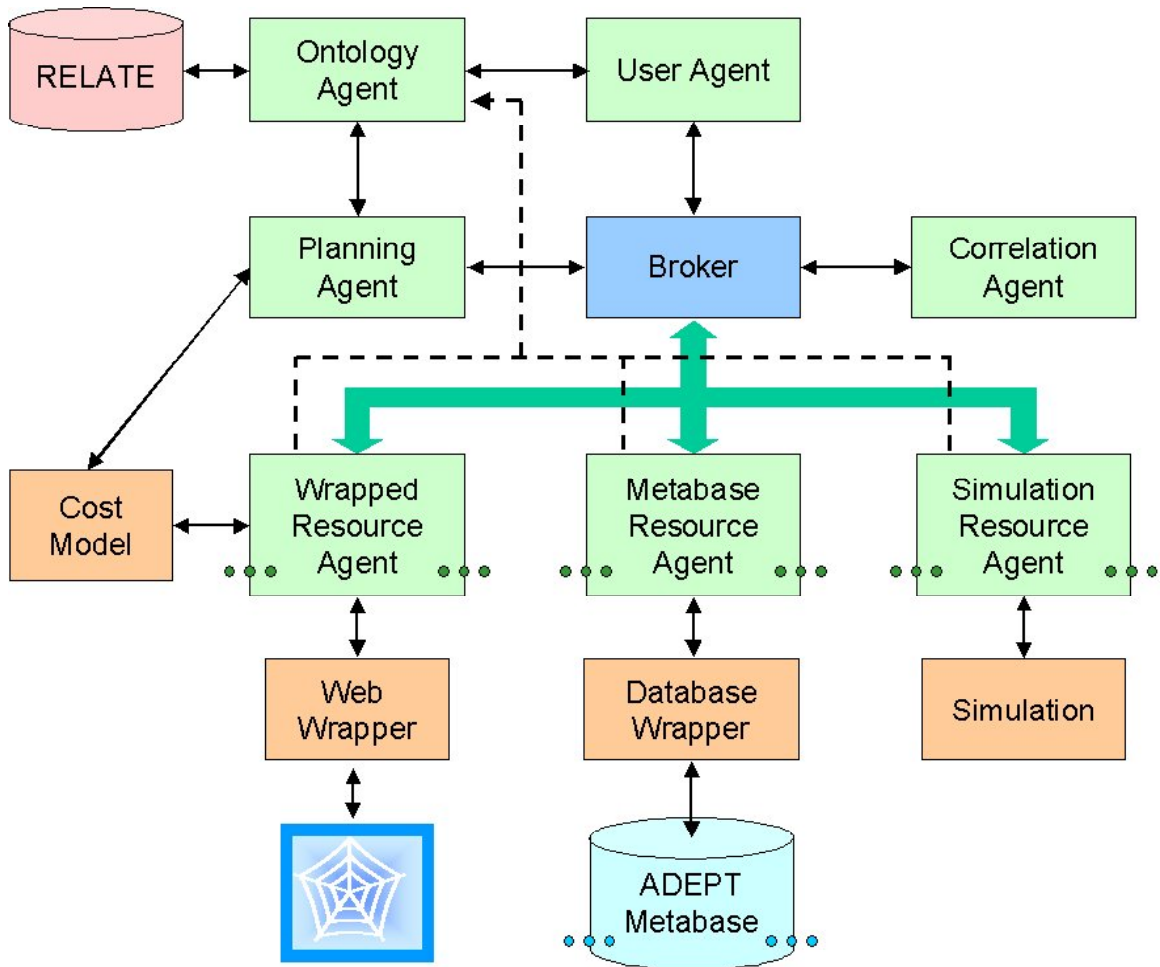


Fig. 4.1 System architecture

4.1.2 THE ONTOLOGY AGENT

The Ontology agent is the core of the semantic interoperability engine. It allows design and management of ontologies in the system and serves ontology information to the different agents when requested. This is also the place where inter-ontological relationships are computed and semantic discrepancies are removed. Section 4.2 discusses this agent in detail.

4.1.3 THE PLANNING AGENT

This agent is responsible for creating an execution plan for the requested Iscape [SIN00]. It decomposes the Iscape into sub-queries, which are specified in a high level language. This hides the details of local data organization and representation of each resource agent.

4.1.4 THE CORRELATION AGENT

The responsibilities of this agent involve integration and merging of the results and elimination of duplicates [SIN00]. It also performs operations like the simulation operation.

4.1.5 THE RESOURCE AGENT

There exists a Resource agent for each data source and database. Each agent manages the resources assigned to it and advises the Ontology agent of the resources held. It translates the global level query given by the Planning agent into resource specific formats, queries the resources and returns the data in the global format. For a more detailed study, refer [GUN00].

4.1.6 THE BROKER

The Broker acts as the central hub to all the other agents. The agents, which may reside elsewhere, communicate with each other through the Broker. It pairs agents seeking a particular service with agents that can perform that service [SIN00]. It provides dynamic

functionality to the system, as agents can come online or go offline without affecting the whole system.

4.2 THE ONTOLOGY AGENT

The Ontology agent helps in achieving semantic inter-operability. It maintains a list of all the ontologies present in the system and provides methods for maintenance operations like addition, deletion and modification. It is responsible for retrieving the ontologies and any other related information if requested by other agents. It directly interacts with the following agents:

- The User Agent: It requires ontology information for Iscape building. It requests a list of ontologies and the various attributes of the different ontologies. This information is used to build and execute Iscapecs interactively.
- The Resource Agent: Every resource when created, publishes itself with the Ontology Agent. It then furnishes the Ontology Agent with the attributes of a particular ontology that it handles. This information is later used by the Ontology agent as described next.
- The Planning Agent: It queries the Ontology agent to find out which Resource agents to contact in order to retrieve the information for that particular ontology.

Each ontology contains the ontology name, keywords associated, the parent ontology, description, the attributes of the ontology and the different resource agents that handle those attributes, as described in section 3.1. The ontologies themselves are stored as XML documents. Methods are then provided to read and write the ontology in XML. There are several advantages of doing this, as described:

- It provides for a standardized schema and document validation when transferred between agents;
- It provides an accurate description of the contents by the use of tags;
- Transfer of data between the agents becomes easier;
- Searching of specific elements within the document becomes easier.

A graphical ontology manager has been developed for user-level administration and allows complete functionality of the Ontology agent in graphical form. Shown in figure 4.2 is a snapshot of the manager, developed using Java Swing.

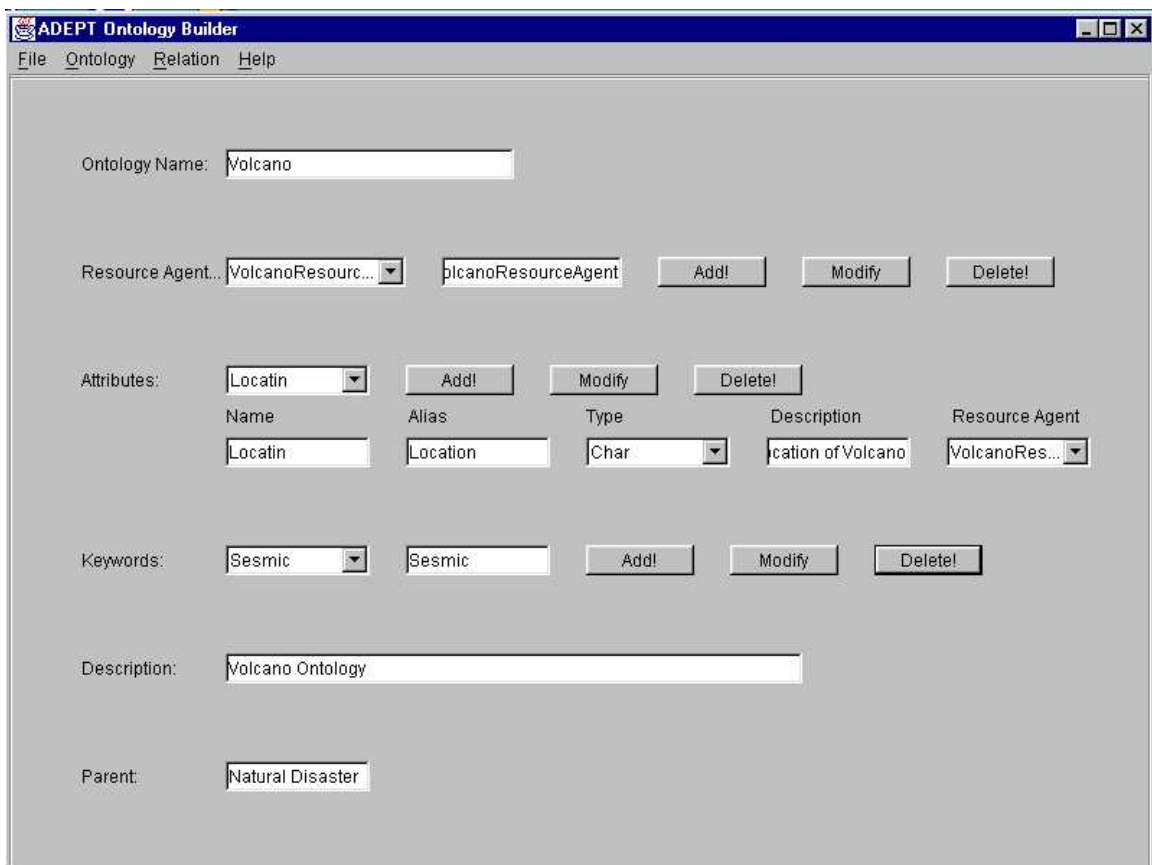


Fig. 4.2 Snapshot of the graphical ontology builder

4.3 THE RELATIONSHIP MANAGER

An information store (RELATE) has been developed which maintains the different relationships created and information about them. The store itself is implemented in

XML, and maintains the following information:

- Schema for different relationships like “affects”;
- The existence of different relationships between the ontologies present;
- The synonym/hypernym/hyponym relations between the ontologies.

Putting together the above information, we can extract several details about a relationship like “A affects B”. We have provided the following methods that can be utilized in this

regard:

- Retrieving all the ontologies affected by A: This can be achieved by parsing the records in the XML store and retrieving all the records that include A as an element in the relation on the left hand side;
- Retrieving all the ontologies that affect B: This operation is similar to the one described above, except for the fact that we have to search records that have B as an element on the right hand side;
- Checking of transitive relationships: For a relation like A affects B; B affects C => A affects C – we need to scan the store to find out related records where both the common “B” ontologies involved have a common attribute set.

A sample relationship between two ontologies described in XML is shown:

```
<?xml version="1.0"?>
<!DOCTYPE Relation SYSTEM "relation.dtd">
  <Relation>
<Description>Affects relationship between Volcano and
  Environment</Description>
  <Construct>
    <Complex>
```

```

    <OntologyA Type="Leaf">Volcano</OntologyA>
      <OntologyB
Type="Leaf">Environment</OntologyB>

        <Expression>
          <OperatorSet>
            <Config>No</Config>
            <AFunction>Time</AFunction>
            <AElement>Volcano</AElement>
            <Operator>EQ</Operator>
            <BFunction>Time</BFunction>
            <BElement>Environ</BElement>
          </OperatorSet>
          .
          .
          <RelationSet>

<AElement>PyroclasticFlows</AElement>
  <SubRelation>Kills</SubRelation>
  <BElement>People</BElement>
  </RelationSet>

  <RelationDesc>Pyroclastic flows of
volcanoes kill people if people come across
it</RelationDesc>
  </Expression>
  .
  .
  .
  .
  </Complex>
</Construct>
</Relation>

```

We now describe how the calculations of relationship expressions discussed in Section 3.4 are implemented. Let us recall how the relationship expression looks like:

$$[FN(x) \textit{ operator} FN(y)]^* \Rightarrow [ASC \textit{ relation} BSC]$$

We can see that there are two entities that need deeper functionality – the enclosing function FN and the *operator*. Let us first discuss the working of the enclosing function.

This has been implemented as a method that takes an ontology name or any of its attributes (e.g., Ontology.Attribute) and returns a value of a specific type that has been predefined in the system. For example, Weight (Pizza.Cheese) would always return a

number in a specific format and predefined unit (say x.xx grams), regardless of how the value in the ontology was. Mapping functions are provided to this effect. This eliminates heterogeneity in formats and units, and provides a homogeneous interface for the operator to work on. The set of enclosing functions and mappings may differ from domain to domain. We have developed a set of functions for the natural disasters domain. However, the schema developed is generic and the system will work for any domain provided the specific functions are plugged in. These functions can be written by the user at any time.

Let us now discuss the operator. As seen from the examples earlier, the operator takes two parameters (or enclosing functions) and returns a Boolean value depending on the result of the evaluation. The two functions should have return values of the same type or else comparison is not possible. The syntax for calling the operator method is as shown:

EQUAL (Volume (Ash Rain) , Value (50))

INTERSECT (Location (Volcano), Location (Environment))

The method then performs the required operation on the different values, providing a tolerance depending on the granularity of the values supplied. For example, in case of comparing the location of the volcano, a one mile inaccuracy may not matter much, whereas in the case of a building, not more than 10 meters in tolerance is acceptable.

These granularity values, which are highly entity dependant, are stored in an XML properties file. The method consults the file before the comparison operation and takes into account the acceptable tolerance for the entity at hand. This results in a kind of fuzzy match. Of special interest are the functions involving geo-spatial (Area and Location) and

temporal (Date and Time) values, which are of importance in a geo-referenced digital library based system like ADEPT. Analogous to the enclosing functions, these operator methods can be user-defined for a specific domain.

These methods, which are part of the Relationship manager, are called by the Planning agent when it encounters the relationships in the queries and needs to evaluate them, with the data obtained from the different sources, before it can pass them on to the Correlation agent.

A graphical builder has also been developed for the creation and modification of relationships. Shown in figure 4.3 is a snapshot of the relationship builder, built using Java Swing.

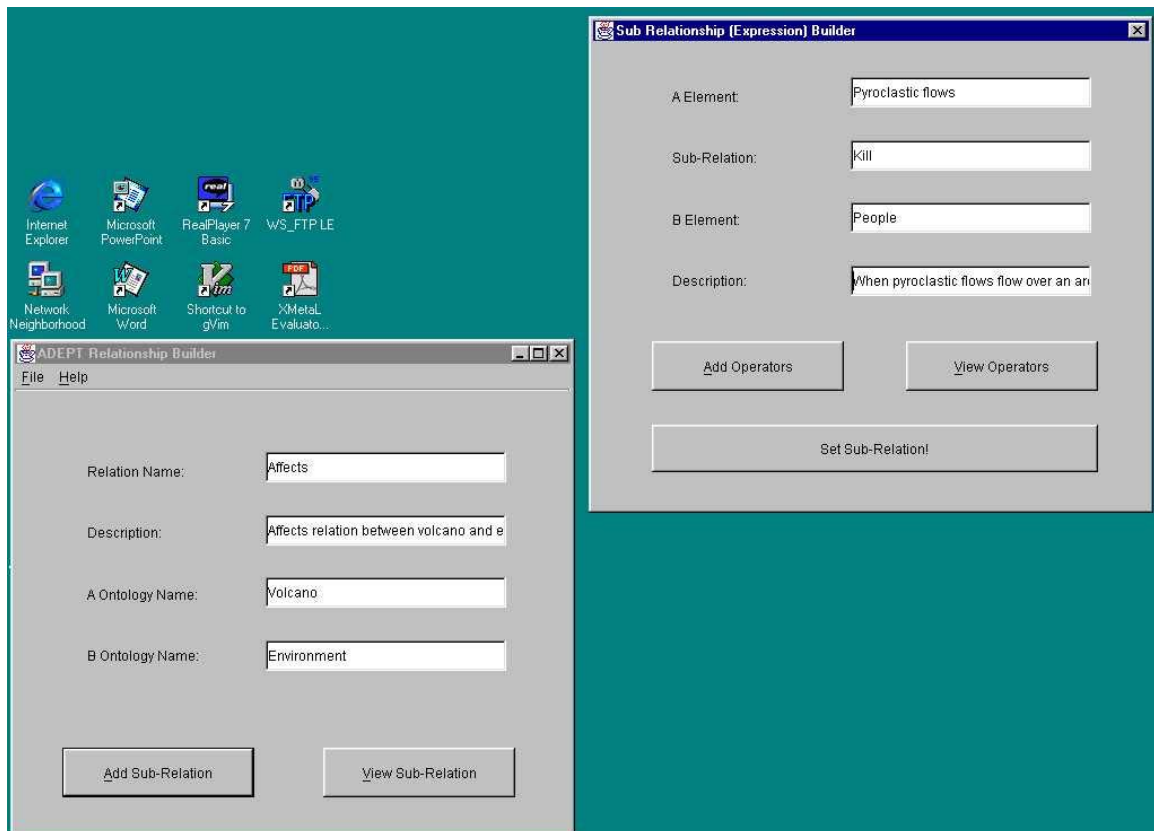


Fig. 4.3 Snapshot of the Relationship builder

4.4 SYSTEM DEMONSTRATION

Shown herein is a walkthrough of the system. Figure 4.4 shows the student user selection of Iscapec. The different agents then come into play. The Iscape is passed on to the planning agent which creates appropriate query plans to execute. These plans are then passed on the resource agent, which executes these queries and returns the data sets to the correlation agent. The correlation agent does a cleanup operation, puts together the data in a homogeneous format and returns the information to the user agent for display. These results are displayed as shown in figure 4.5.

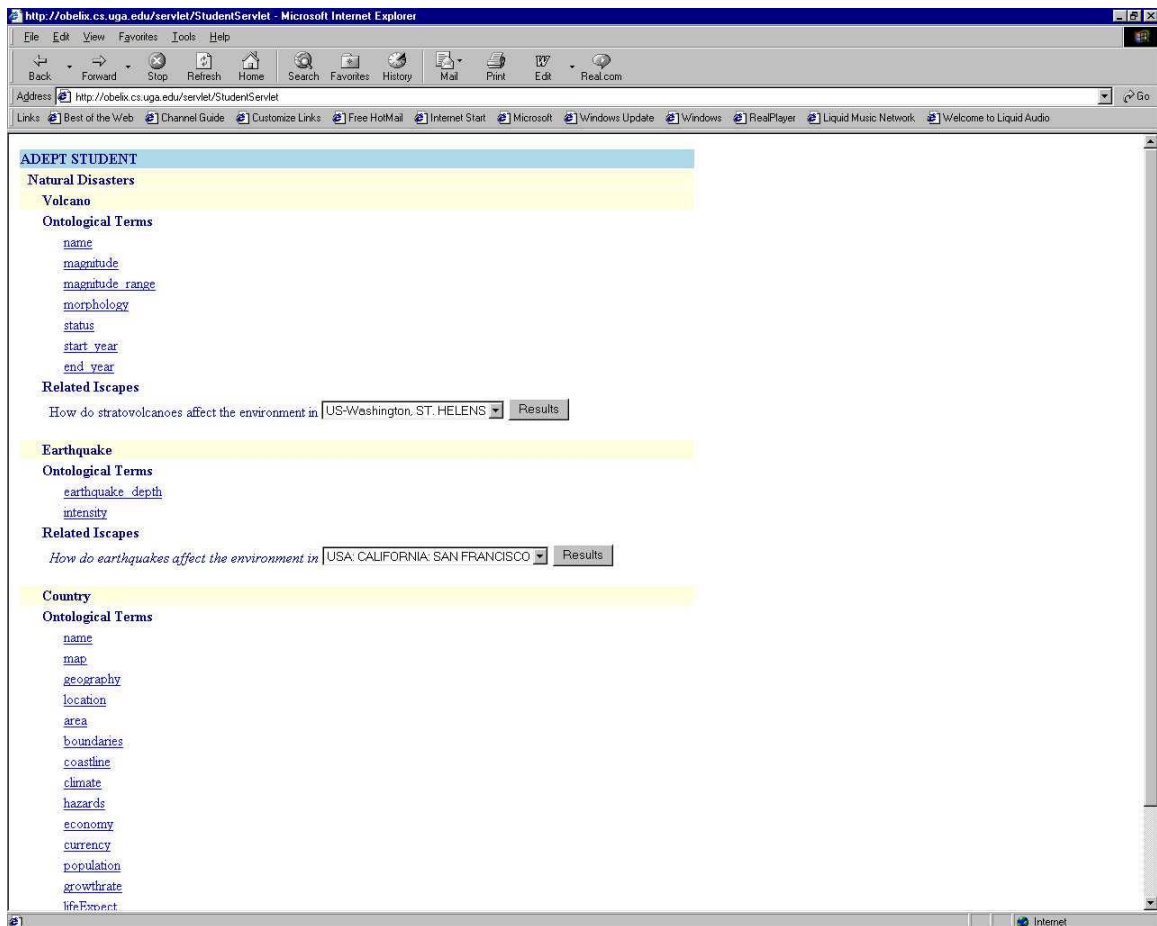


Fig. 4.4 Iscape selection

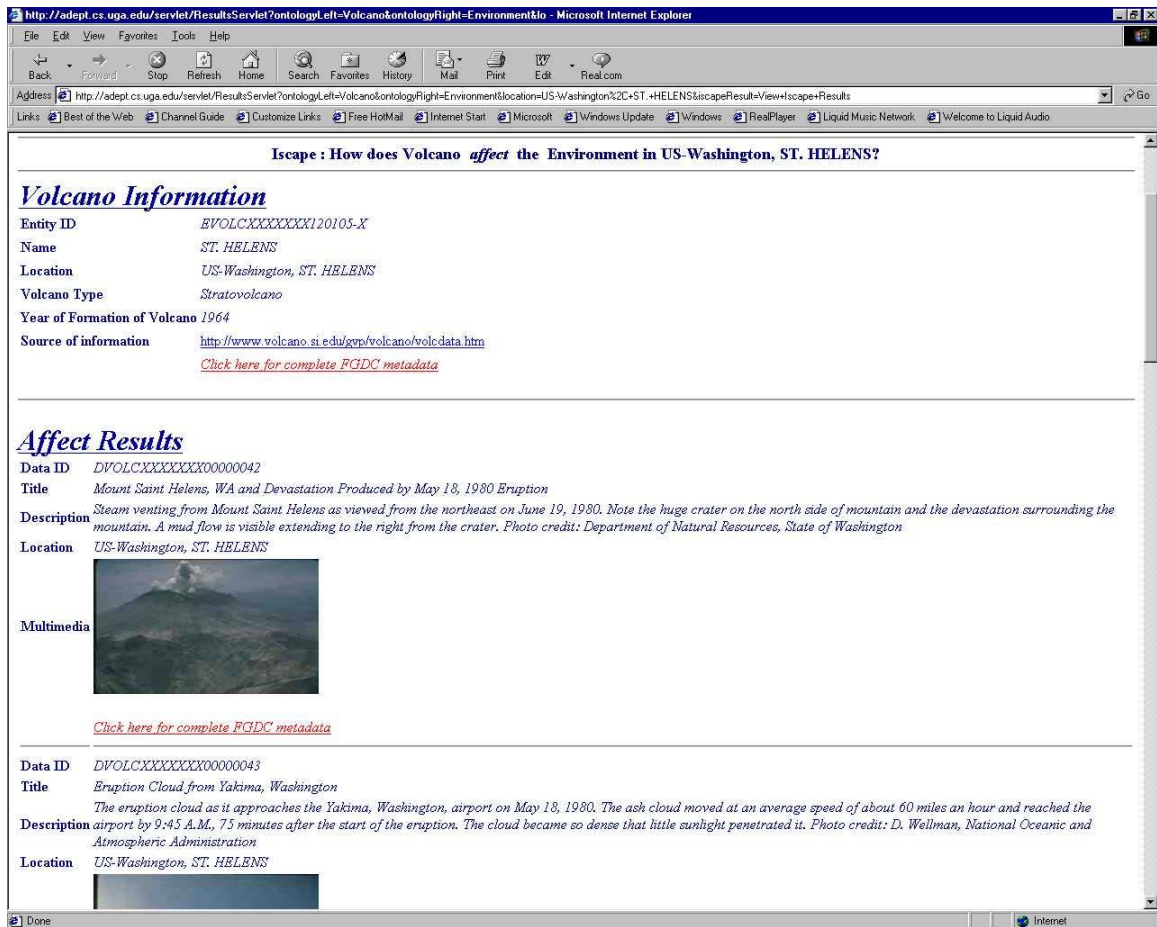


Fig. 4.5 Results

The simulation operation user input is as depicted in figure 4.6. Shown is the population growth simulation for the San Francisco Bay Area using the Clarke Urban Growth Model. Figure 4.7 shows the results of this simulation.

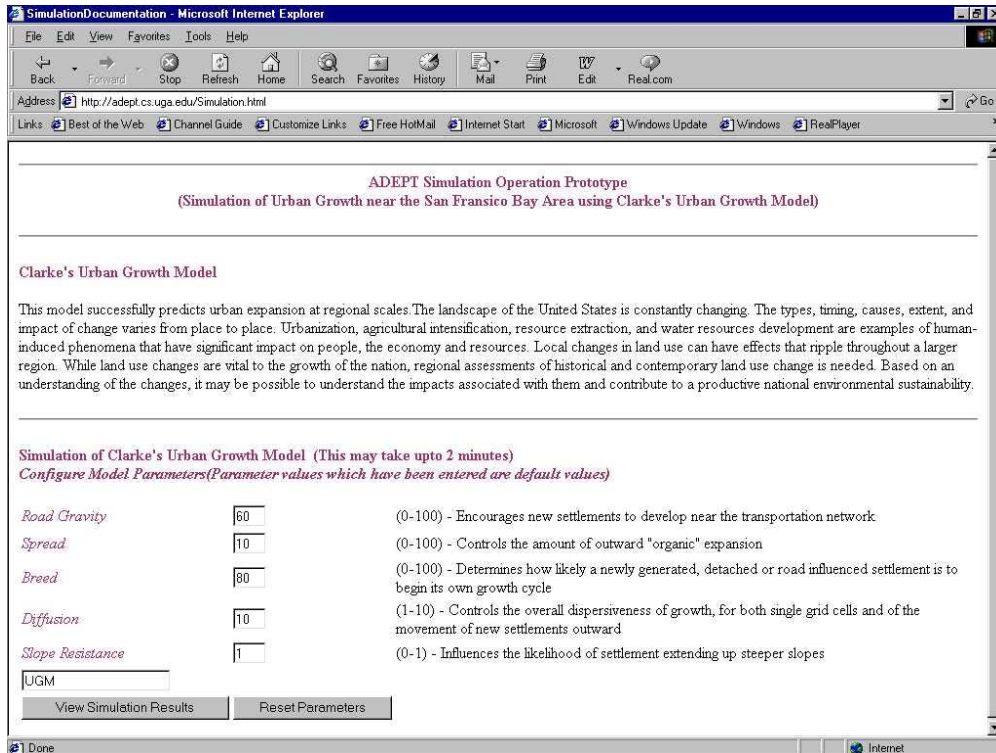


Fig. 4.6 Simulation operation

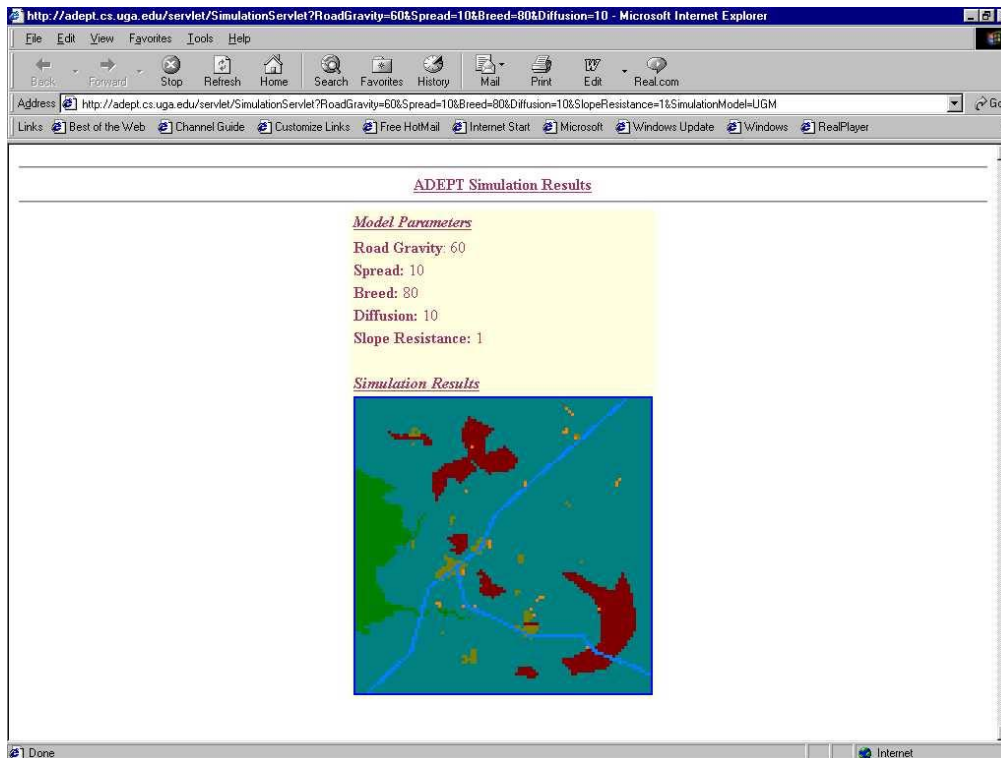


Fig. 4.7 Simulation results

CHAPTER 5

CONCLUSION AND FUTURE WORK

This chapter recaps what was achieved in this thesis. We also discuss additions and improvements that could be done at a later date.

5.1 CONTRIBUTIONS

Semantic heterogeneity is a big obstacle towards realizing the ideal of a Semantic Web.

In this thesis, we have attempted to achieve semantic inter-operability by the use of ontologies and relationships. In particular, we have developed:

- A framework for ontologies and relationships: A set of ontologies for the domain of Natural Disasters has been developed for use in related Iscapecs. An Ontology Agent had been developed that maintains the ontologies and serves the required information to the different agents. A graphical builder for ontologies has been built for user access;
- A scheme for inter-ontological relationships: A scheme for relationship access and management has been developed (RELATE). A Relationship Manager has been developed to this effect, along with a graphical builder. A generalized schema for the “Affects” relationship had been developed and deployed in the ADEPT Iscapecs;
- A prototype system incorporating semantic inter-operability: A mediation-like architecture has been developed, which integrates multi-media data from

heterogeneous sources. The Digital Earth paradigm has been modeled to provide a learning model for the geography student.

5.2 FUTURE WORK

No work is ever complete. There are always lots of enhancements and related work that can be done. Below are some of them:

- Creating a knowledge base using a bigger relationship store, that can be used to answer more diverse information requests;
- Building of a full-fledged integrated user interface (portal) to manage all aspects of semantics, including ontologies and relationships;
 - Relationships involving more than two ontologies;
- Relationships that are completely scalable and that can be used as a sub-module in bigger systems incorporating a wider scope.

BIBLIOGRAPHY

- [ACH93] Arens Y., Chee C., Hsu C. and Knoblock C. "Retrieving and Integrating Data from Multiple Information Sources". *International Journal of Intelligent and Cooperative Information Systems*, 1993.
- [ADE99] "The Alexandria Digital Earth Modeling System (ADEPT)". Project Proposal.
- [BCB00] Bornhovd C. and Buchmann A. P. "A Prototype for Metadata-based Integration of Internet Sources".
- [BER98] Bertram C. "Semantic Correlation of Heterogeneous Distributed Assets in InfoQuilt". *Masters Thesis*. 1998.
- [BER99] Berners-Lee T. "Weaving the Web". *Harper San Francisco*, 1999.
- [BKV97] Bayardo R. J., Bohrer W., Cichocki A., Flower G., Helal A., Kashyap V., Ksiezyk T., Martin G., Nodine M., Rashid M., Shea R., Rusinkiewicz M., Unnikrishnan C., Unruh A. and Woelk D. "InfoSleuth: Agent-Based Semantic Integration of Information in Open and Dynamic Environments". *Proceedings of the 1997 ACM International Conference on the Management of Data*. 1997.
- [BOR00] Bornhovd C. "Semantic Metadata for Integration of Web-based Data for Electronic Commerce".
- [BRI00] "Encyclopedia Britannica". <http://www.britannica.com>

- [CGH94] Chawathe S., Garcia-Molina H., Hammer J., Ireland K., Papakonstantinou Y., Ullman J. and Widom J. "The TSIMMIS Project: Integration of Heterogeneous Information Sources", *Stanford University*, 1994.
- [CHE99] Chen H. "Semantic Research for Digital Libraries". *D-Lib Magazine*, Oct 1999.
- [COL97] Colomb R. M. "Impact of Semantic Heterogeneity on Federating Databases". *The Computer Journal*. 1997.
- [EQE00] "How an Earthquake Causes Damage".
<http://www.worldbook.com/fun/bth/earthquake/html/howdamaged.htm>
- [FFL00] Fuzzy Sets and Fuzzy Logic. <http://members.aol.com/btluke/fuzzy01.htm>
- [FMR00] "Fuzzy Matching as a Retrieval-Enabling Technique for Digital Libraries". www.asis.org/midyear-96/girillpaper.html
- [FSO00] "Fuzzy Sets and Operations". http://www-dse.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/sbaa/report.fuzzysets.html
- [FUZ00] "Overview of Fuzzy Concepts".
http://ai.iit.nrc.ca/IR_public/fuzzy/fuzzyJavaDocs/overview.html
- [FZL00] "Fuzzy Logic". <http://www.personal.kent.edu/~jtboehm/fuzzy.html>
- [GEN92] Genesereth M. R. and Fikes R. E. "Knowledge Interchange Format. Stanford", 1992.
- [GGL00] Google. <http://www.google.com>.
- [GRU00] Gruber T. "What is an Ontology?". <http://www-ksl.stanford.edu/kst/what-is-an-ontology.html>

- [GUN00] Guntamadugu M. “Automating Metabase Creation from Multiple Heterogeneous Sources”. *Masters Thesis, University of Georgia*, 2000.
- [HHV96] Heiler S., Miller R. J., Ventrone V. “Using Metadata to Address Problems of Semantic Interoperability in Large Object Systems”. <http://www.computer.org/conferences/meta96/rmiller/paper.html>, 1996.
- [HVE00] “Hazardous Volcanic Events”.
<http://magic.geol.ucsb.edu/~fisher/hazards.htm>
- [KAS94] Kashyap V. and Sheth A. “Semantics-based Information Brokering: A step towards realizing the Infocosm”. *CIKM94*. 1994.
- [KAS99] Kashyap V. “Design and Creation of Ontologies for Environmental Information Retrieval”. *KAW99*. 1999.
- [KAS00] Kashyap V. and Sheth A. “Information Brokering Across Heterogeneous Digital Data: A Metadata-Based Approach”. *Kluwer Academic Publishers*, 2000.
- [LEX00] “Lexical Semantic Relations”.
<http://www.ilc.pi.cnr.it/EAGLES96/rep2/node7.html>
- [MAH99] Mahajan V. “A Multi-Agent System for Metadata Management and Information Brokering”. *Masters Thesis, University of Georgia*, 1999.
- [MAL00] Maluf D. A. and Wiederhold G. “Abstraction of Representation for Interoperation”.
- [MIT00] Mitra P., Wiederhold G. and Kersten M. “A Graph-Oriented Model for Articulation of Ontology Interdependencies”. Stanford University. 2000.
- [MKS96] Mena E., Kashyap V., Sheth A. and Illarramendi A. “OBSERVER: An approach for query processing in global information systems based on interoperation across pre-existing ontologies”. *Proceedings of the First IF-CIS International Conference on Cooperative Information Systems*, 1996.

- [NDH00] “Natural Disasters and Hazards”.
http://dmoz.org/Science/Earth_Sciences/Natural_Disasters_and_Hazards
- [NGD00] “National Geophysical Data Center (NGDC)”. <http://www.ngdc.noaa.gov>
- [ONT00] “Ontolingua”. Stanford University. <http://ontolingua.stanford.edu>
- [PAL00] Palsena N. “A Collaborative Approach to Learning using Information Landscapes”. *Masters Thesis, University of Georgia, 2000.*
- [RAY99] Ray S. “Ontology-based Media Independent Correlation of Information Across Heterogeneous Distributed Sources”. *Masters Thesis, University of Georgia, 1999.*
- [RLR00] ”The Relational Repository”,
<http://www.c3.lanl.gov/~rocha/lww/RelRep.html>
- [REL00] “Relations”. <http://osm7.cs.byu.edu/OSA/relation.html>
- [RER99] Rodriguez M. A., Egenhofer M. J., Rugg R. D. “Assessing Semantic Similarities Among Geo-spatial Feature Class Definitions”. *INTEROP-99, 1999.*
- [SHE98] Sheth A. P. “On Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics”. *In Interoperating Geographic Information Systems, 1998.*
- [SEM00] “Semantic Properties and Relations”.
http://www.arts.uwa.edu.au/LingWWW/LIN101-102/NOTES-101/truth_relations.html

- [SET00] “Sets, Relations”.
<http://www.cl.cam.ac.uk/Teaching/1998/IntroAlgs/notes98/node4.html>
- [SIN00] Singh D. “An Agent Based Architecture for Query Planning and Cost Modeling of Web Sources”. *Masters Thesis. University of Georgia*, 2000.
- [STA00] Staab S., Angele J., Decker S., Erdmann M., Hotho A., Maedche A., Schnurr H., Studer R. and Sure Y. “Semantic Community Web Portals”. *University of Karlsruhe*. 2000.
- [VOL00] “How do Volcanoes affect the environment?”.
http://volcano.und.nodak.edu/vwdocs/frequent_questions/grp12/question588.html
- [WBP00] Weinstein P. C. and Birmingham W. P. “Comparing Concepts in Differentiated Ontologies”.
<http://www-personal.umich.edu/~peterw/KAW99/weinstein.html>
- [WCD00] “What Causes Damage?”.
http://www.geophys.washington.edu/SEIS/PNSN/INFO_GENERAL/NQT/what_causes_damage.html
- [WIE92] Wiederhold G. “Mediators in the Architecture of Future Information Systems”. *IEEE Computer, Vol.25 No.3*, March 1992, pages 38-49.
- [WEI94] Wiederhold G., “Interoperation, Mediation, and Ontologies”. *Stanford University*. 1994.
- [WIE97] Wiederhold G. “Scalable Knowledge Composition”. *Stanford University*, 1997.
- [YAH00] Yahoo! <http://www.yahoo.com>.

APPENDIX A
ONTOLOGY DTD

```
<!-- XML DTD for the Ontology -->
<!-- @author Sriram Lakshminarayan -->

<!ELEMENT Ontologies (Ontology)*>
<!-- Components of the Ontology -->
<!ELEMENT Ontology (Name, Keywords, Description, Attributes,
Parent, ResourceAgents)>

<!ELEMENT Keywords (Keyword)*>
<!ELEMENT Attributes (Attribute)*>
<!ELEMENT ResourceAgents (Agent)*>

<!ELEMENT Agent (Attrib)*>
<!ATTLIST Agent
    Name CDATA          #REQUIRED>

<!ELEMENT Attribute      (#PCDATA)>
<!-- Different components of the 'Attribute' -->
<!ATTLIST Attribute
    Name      CDATA          #REQUIRED
    Alias     CDATA          #REQUIRED
    Type      CDATA          #REQUIRED
    Desc      CDATA          #REQUIRED>

<!ELEMENT Name           (#PCDATA)>
<!ELEMENT Keyword        (#PCDATA)>
<!ELEMENT Description    (#PCDATA)>
<!ELEMENT Parent         (#PCDATA)>
<!ELEMENT Attrib         (#PCDATA)>

<!-- End of DTD -->
```

APPENDIX B

RELATIONSHIP DTD

```
<!-- XML DTD for the AFFECTS relationship -->
<!-- @author Sriram Lakshminarayan -->

<!ELEMENT Relation (Description, Construct)>
<!ELEMENT Description (#PCDATA)>
<!-- Three types of relations possible -->
<!ELEMENT Construct (Simple | Complex | Transitive)>

<!-- Ontology A as a whole affects a set of elements of B --
>
<!ELEMENT Simple (OntologyA, OntologyB, SetB*)>
<!ELEMENT SetB (SubRelation, BElements, RelationDesc)>

<!-- Some/each element in A affect a set of elements of B --
>
<!-- Config says Yes/No to BElement being configurable -
this is used in the Iscape -->
<!ELEMENT Complex (OntologyA, OntologyB, Expression*)>
<!ELEMENT Expression (OperatorSet*, RelationSet,
RelationDesc)>
<!ELEMENT OperatorSet (AFunction, AElement, Operator,
BFunction, BElement, Config)>
<!ELEMENT RelationSet (AElement, SubRelation, BElement)>
<!ELEMENT AFunction (Name, Attribs)>
<!ELEMENT BFunction (Name, Attribs)>
<!ELEMENT Attribs (Attrib)*>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Config (#PCDATA)>

<!-- A affects B; B affects C => A affects C -->
<!ELEMENT Transitive (Relations)*>
<!ELEMENT Relations (OntologyA, AAttribs, OntologyB,
BAttribs)>
<!ELEMENT AAttribs (Attrib)*>
<!ELEMENT BAttribs (Attrib)*>
<!ELEMENT Attrib (#PCDATA)>
```

```
<!-- Each ontology can be a Leaf or Root -->
<!ELEMENT OntologyA (#PCDATA)>
<!ATTLIST OntologyA Type CDATA #REQUIRED>

<!ELEMENT OntologyB (#PCDATA)>
<!ATTLIST OntologyB Type CDATA #REQUIRED>

<!ELEMENT Operator (#PCDATA)>
<!ELEMENT SubRelation (#PCDATA)>
<!ELEMENT RelationDesc (#PCDATA)>
<!ELEMENT AElement (#PCDATA)>
<!ELEMENT BElement (#PCDATA)>

<!-- End of DTD -->
```