ENABLING SEMANTIC MAPPING
IN
WEB SERVICES MODELING FRAMEWORK

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Computer Science and Engineering

Thapar University, Patiala

By:
DEEPTI
(80632004)

Under the supervision of:
Ms. Shalini Batra
Sr. Lecturer

MAY 2008

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "Enabling Semantic Mapping in Web Services Modeling Framework", in partial fulfillment of the requirements for the award of degree of Master of Engineering in Computer Science and Engineering submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Ms. Shalini Batra and refers other researcher's works which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

(***Deepti)***

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(**Ms. Shalini Batra**)
Computer Science and Engineering Department
Thapar University
Patiala

Countersigned by

(SEEMA BAWA)
Professor & Head
Computer Science & Engineering. Department
Thapar University
Patiala

(R.K.SHARMA)
Dean(Academic Affairs)
Thapar University,
Patiala.

ACKNOWLEDGEMENT

The thesis work on Web Service Modeling Ontology (WSMO), though a humble and average effort on my part, would not have been possible of realization without the active guidance and willing support of my supervisor '**Ms. Shalini Batra', Sr. Lecturer, TU, Patiala.** She, not only provided me timely guidance with regards to clarity of concepts and involved ideas, but also helped me in the collection of material/ books/ articles etc. on the subject. I am thankful for her continual support, encouragement, and invaluable suggestions throughout my degree. No amount of words that I may use to thank her would sufficient enough to express my sense of gratitude to her.

I am also beholder to **'Ms. Seema Bawa', Head, Computer Science and Engineering Department, TU, Patiala** for creating an atmosphere highly congenial to studies and research. I will be failing in my duty if I don't express my gratitude to **'Dr. Abhijit Mukherjee', Director of the University,** for making provisions of infrastructure such as library facilities, computer labs equipped with net facilities, immensely useful for the learners to equip themselves with the latest in the field.

I am also thankful to my fellow students at TU, who rendered unstinted help to me as and when I sought their help.

I am emotionally thankful to my parents. They give importance to education and values more than anything else in life. I 'Thank you' papa and mom and the word THANK YOU is not sufficient to express what I feel for you, words unsaid are the fonder than the ones said. I will try to fulfill your dreams that you have cherished for your children i.e. to make a niche in the field of interest and studies.

I am equally emotionally thankful to my husband Dr. Sunil and my educationist in-laws, who uphold the importance of education in life. They provided me all the facilities, time and emotional support to complete the course successfully.

DEEPTI GUPTA

ABSTRACT

Web services and service-oriented architectures share a common goal of making software components and business applications available via standardized interfaces, within or across enterprise boundaries. Web services aim to employ the Web as a global infrastructure for distributed computation, for integrating various applications, and for the automation of business processes. The Web will be not only the place where human-readable information is published, but also the place where global computing is realized. The Semantic Web is a vision of a new architecture for the World Wide Web, characterized by the association of machine-accessible formal semantics with traditional Web content. Web Services and the Semantic Web are transforming the World-Wide Web from a large collection of information into an intelligent distributed computational device. The Semantic Web allows the representation and exchange of information in a meaningful way. Adding machine-processable semantics to web services bring Services Oriented Architectures to their full potential as critical subtasks can be automated by integrating web services with Semantics. This paradigm shift in the intended purpose of the Web, calls for transforming the Web from being people-centric to software applications-centric. It leaves humans to focus on problem only.

The emerging concept of Semantic Web Services (SWS) develops techniques for the automated discovery, composition, and execution of Web services. The technology of Semantic Web Services envisions the easy access to various systems, seamless integration of heterogeneous entities, ad-hoc cooperation between various business communities and dynamic collaborations on the Web. The Semantically Enabled Service-Oriented Architectures (SESA) aims at automated support for the complete life-cycle and production cycle of Service-Oriented Architectures. For this, the SWS approach is extended with additional, semantically described elements: goals as formally described client objectives, and mediators for handling potentially occurring heterogeneities. The Web Service Modeling Ontology (WSMO) is a comprehensive framework for SESA that defines a conceptual model along with a specification language, and provides a reference implementation with additional tool support. WSMO considers four main elements; these

are ontology, goal, web service, mediators. Web Service Modeling Execution (WSMX) acts as middleware between the client and endpoint Web Services. Semantic Execution Environment creates an environment capable of managing all the aspects related to semantically enhanced Web services, to enable their discovery, selection, mediation and execution.

In this thesis the intent is to work on web service toolkit, which provides various tool supports for WSMO. Ontologies are viewed in graphical form. Both services and goals are conceptualized as ontologies. Queries are made to the ontologies by WSML reasoner. Mappings are done with both inbuilt and imported ontologies and mapping results are taken. Goals and web services are added and discovered through WSMX.

TABLE OF CONTENTS

CERTIFICATE ……..… ii
ACKNOWLEDGEMENT …… iii
ABSTRACT ……..... iv
TABLE OF CONTENTS ……..... vi
LIST OF FIGURES ……..... viii

CHAPTER 1: INTRODUCTION ……... 01

1.1 MOTIVATION ……... 01
1.2 ORGANIZATION OF THESIS …… 02

CHAPTER 2: BACKGROUND INFORMATION …….. 03

2.1 SEMANTIC WEB …….. 03

2.2 FRAMEWORKS FOR SEMANTIC WEB ……. 05

2.2.1 OWL-S ……..... 05

2.2.2 WSDL ……..... 06

2.3 WEB SERVICE MODELING ONTOLOGY ……..... 06

2.3.1 ONTOLOGY ……...…. 08

2.3.1.1 NEED OF ONTOLOGY ……..... 10

2.3.1.2 USE OF ONTOLOGY ……..... 10

2.3.2 GOALS …….. 11

2.3.2.1 GOAL SPECIFICATION ……...…. 12

2.3.3 WEB SERVICE …… 13

2.3.3.1 WEB SERVICE ARCHITECTURE …… 13

2.3.3.2 WEB SERVICE OPERATIONS ……. 14

2.3.3.3 FUNCTIONAL ASPECT …….. 15

2.3.3.4 BEHAVIORAL ASPECT ……. 16

2.3.4 MEDIATORS ……..... 16

2.4 WEB SERVICE EXECUTION ……. 17

2.5 WEB SERVICE MODELING TOOLKIT …… 20

CHAPTER 3: LITERATURE REVIEW …… 21

CHAPTER 4: PROBLEM STATEMENT …….. 24

4.1 PROBLEM DEFINITION …….. 24

4.2 METHODOLOGY ……. 25
CHAPTER 5: IMPLEMENTATION ……..26
5.1 BROWSING AND EDITING ONTOLOGIES USING WSMT …….26
5.1.1 ONTOLOGIES VIEWS ……...26
5.1.2 IMPORTING ONTOLOGIES ……28
5.1.3 QUERYING ONTOLOGIES ……..29
5.2 MAPPING ONTOLOGIES ……... 30
5.3 SEMANTIC EXECUTION ENVIRONMENT VIEW ……..32
5.4 WSMX ……..33
CHAPTER 6: CONCLUSION AND FUTURE SCOPE …….35
6.1 CONCLUSION ……...35
6.2 FUTURE SCOPE ……...37
REFERENCES ……38
APPENDIX ……...41
LIST OF PUBLICATIONS ……...42

LIST OF FIGURES

Figure 2.1 **Four Elements of WSMO …….. 08**

Figure 2.2 **Ontology Definition ……..……..... 09**

Figure 2.3 **Client and Server Interaction to Provide Web Services …..... 14**

Figure 2.4 **Discovery of Desired Goal ……. 18**

Figure 2.5 **WSMX Architecture …….. 19**

Figure 5.1 **WSML Text Editor View …….. 27**

Figure 5.2 **Graphical View of Ontology in WSMT ……... 27**

Figure 5.3 **Form Based Editor ……28**

Figure 5.4 **Importing File in WSML Perspective …….. 29**

Figure 5.5 **WSML Reasoner Result after Querying …….. 29**

Figure 5.6 **Creating Mapping File …….. 30**

Figure 5.7 **Generation of Inferences ……... 31**

Figure 5.8 **Mapping Result …….. 31**

Figure 5.9 **SEE View …… 33**

Figure 5.10 **WSMX Page……... 33**

Figure 5.11 **Discovery View in WSMT ……... 34**

Chapter 1
INTRODUCTION

1.1 MOTIVATION

Computers and computer networks are used mainly for storing and sending information, and the interpretation of this information is done by software engineers and domain experts. Existing technologies enabling the integration of enterprise systems use very few capabilities of modern computers. Web Services have promised to solve some of these problems. Web Services will transform the web from a collection of information into a distributed computational device. But because of their syntactical nature, they have failed in most of these cases and humans must still be kept in the loop. Web Services are self-contained, self-describing, modular applications that can be published, located, and invoked over the Web. This definition, like any of many such definitions describing Web Services, makes no comment on who should publish, locate and invoke them. The hidden answer is that these are the humans, who are involved in almost every step of Web Services usage process. [1].

The unquestionable success of existing Web Service specifications lies in their ability to separate service interface from its implementation, based on standards which are accepted by all the major players of the IT industry. However these standards lack an appropriate semantic framework. The application of semantics to Web Services can be used to remove humans from the integration jigsaw and substitute them with machines.

In order to employ their full potential, appropriate description means for web services need to be developed. For this purpose, we define a fully-fledged Web Service Modeling Framework (WSMF) that provides the appropriate conceptual model for developing and describing web services.

1.2 ORGANIZATION OF THESIS

The thesis is divided into six chapters.

Chapter 1- The introduction to the work is provided.

Chapter 2 – The general concepts used in the thesis work are introduced. This chapter serves as theoretical background for the future chapters.

Chapter 3 –Literature review section gives a brief survey of research going on in the area of semantic web and enhancements required there of.

Chapter 4 – After going through literature review problem statement has been identified and defined in this chapter.

Chapter 5 – An evaluation model of work done is described. The illustrations captured during the work done are presented.

Chapter 6 - The conclusion and a summary of all experiences gained during thesis are described.

At the end of thesis, the appendix and references are presented.

Chapter 2

BACKGROUND INFORMATION

2.1 SEMANTIC WEB

The Semantic Web and Web Services are viewed as the enabling technologies for the next generation of web applications. The openness of the web means the volume of published information is growing exponentially, resulting in information overload. The only obstacle to gaining access to this information is a communication link, and finding specific data in this sea of information becomes increasingly difficult. Web Services connect computers and devices with each other using the Internet to exchange data and combine data in new ways. Web Services can be defined as software objects that can be assembled over the Internet using standard protocols to perform functions or execute business processes.

The objective of Web Services is to enable distributed computation over the Internet by automated and dynamic discovery, composition, and execution of services. Web services are a new breed of Web application. Web services enable transparent, language-independent and platform-independent access to services. Web service is a collection of protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks, in a manner similar to inter-process communication on a single computer. They are self- contained: that is one web service can provide full functionality in its domain, self-describing, modular applications: that is once a web service is created it can be invoked by other web services too, that can be published, located, and invoked across the Web. Web services perform functions, which can be anything from simple requests to complicated business processes. Once a Web service is deployed, other applications (and other Web services) can discover and invoke the deployed service. The current Web Service technology stack enables the exchange of messages between Web Services through Simple Object Access Protocol (SOAP), describes the technical interface for consuming a Web Service via Web Service Description Language (WSDL), and supports the advertisement of Web Services

in registries through Universal Description, Discovery, Integration (UDDI).[2] However, these technologies do not include explicit descriptions of the functionality of a Web Service. Moreover, the existing descriptions are represented syntactically and therefore do not depict the meaning of the information to be interchanged.

Semantic Web Services applies Semantic Web technology to Web Services raising the level of discourse. The Semantic Web is an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation. It is not a separate Web but an augmentation of the current one. These apply inference-based reasoning techniques that work on formal descriptions and use ontologies as the underlying data model.[3]. Semantic Web services aim at an integrated technology for the next generation of the Web by combining Semantic Web technologies and Web services, thereby turning the Internet from an information repository for human consumption into a world-wide system for distributed Web computing.

Ontologies are formalized knowledge models, and are propagated as the base technology for the Semantic Web in order to facilitate semantically enabled processing of Web Content. Ontologies define formal shared descriptions of the things that exist in particular domains of interest as well as the relationships that exist between those things. Formal because the descriptions it contains must have a precise provable meaning, and shared as ontology is only valid if its definitions are accepted by a community of users. Ontologies by themselves are static sources of knowledge but become very powerful instruments when combined with logic and reasoning. [1, 4]

The concept of Semantically Enabled Service-Oriented Architectures (SESA) aims at automated support for the complete life-cycle and production cycle of Service-Oriented Architectures (SOA). For this, the SWS approach is extended with additional, semantically described elements: goals as formally described client objectives, and mediators for handling potentially occurring heterogeneities. The Web Services Modeling Ontology (WSMO) initiative provides a complete framework enhancing syntactic description of Web Services with semantic metadata.

2.2 FRAMEWORKS FOR SEMANTIC WEB

Various frameworks for semantic web services are provided. These are: OWL-S, WSDL-S, WSMO. OWL-S and WSMO adopt a similar view on the need of ontologies and explicit semantics but they rely on different logics.

2.2.1 OWL-S

OWL-S has been an initiative of the Semantic Web community to enable automatic discovery, invocation, composition, interoperation and monitoring of Web services through their semantic descriptions. The top-most concept in OWL-S is Service and this in turn consists of three sub-concepts – Service Profile, Service Model and Service Grounding: each characterized by the question it answers.

a) **Service Profile:** It answers: *what does the service provide for prospective clients?* It is used to advertise the service. The intention is to allow an agent to determine whether the service meets its needs. The most essential information presented in the profile, however, is the specification of what functionality the service provides. Information transformation is represented by inputs and outputs; the change in the state of the real world caused by the execution of the service is represented by preconditions and effects. Inputs and outputs refer to OWL classes describing the types of instances to be sent to the service and the respective responses to be expected. The format of the preconditions and effects is not fixed.

b) **Service Model:** It answers: *how is it used?* It tells a client how to ask for the service and what happens when the service is carried out. The main use of a service model is to enable invocation, enactment, composition, monitoring, and recovery. This part of the OWL-S description is responsible for specifying the service interaction protocol in terms of the messages that should be exchanged with the service and the control flow of that exchange.

c) **Service Grounding:** It answers: *how does one interact with it?* It specifies the details of how an agent can access a service. Typically a grounding specifies a

communication protocol, message formats and other service-specific details, such as port numbers.[1,25]

2.2.2 WSDL-S

WSDL-S is an evolutionary and compatible upgrade of the existing WSDL descriptions. WSDL-S is a lightweight approach for adding semantics to Web Services. It allows semantic representation of inputs, outputs, preconditions and effects of Web Service operations, by adding extensions to WSDL. This method keeps the semantic model outside WSDL, making the approach impartial in relation to any ontology representation language. This allows Web service developers to annotate their Web services with their choice of the ontology language. The advantage of this approach is that it builds upon and stays close to existing industry standards. Users can describe both the semantics and operation level details in WSDL - a language with which the developer community is familiar with. [1]

2.3 WEB SERVICE MODELING ONTOLOGY

The Web Service Modeling Ontology (WSMO) is a comprehensive framework for SESA that defines a conceptual model along with a specification language. Having the Web Service Modeling Framework (WSMF) as basis this framework is refined and extended through a formal ontology and language. Web Service Modeling Framework provides a rich conceptual model for the development and the description of web services. WSMF consists of four main elements for describing semantic Web services:

(1) **Ontology** that provide the terminology used by other elements,

(2) **Goals** that state the intentions that should be solved by Web services,

(3) **Web services** descriptions that define various aspects of a Web service, and

(4) **Mediators** which resolve interoperability problems. [5]

WSMO provides ontological specifications for the core elements of Semantic Web services. Appropriate frameworks for Semantic Web services need to integrate the basic Web design principles, those defined for the Semantic Web, as well as design principles for distributed, service-oriented computing of the Web.

WSMO considers three levels of mediation

-Data Level (to mediate heterogeneous Data Sources),

-Protocol Level (to mediate heterogeneous Communication Patterns) and

-Process Level (to mediate heterogeneous Business Processes).

The Web Services Modeling Ontology (WSMO) initiative provides a complete framework enhancing syntactic description of Web Services with semantic metadata. WSMO provides service descriptions for describing services that are requested by service requesters, provided by service providers, and agreed between service providers and requesters. [1] The WSMO project is an ongoing research and development initiative aiming to define a complete framework for SWS and consisting of three activities:

- WSMO, which provides formal specification of concepts for Semantic Web Services,
- WSML (Web Services Modeling Language), which defines the language for representing WSMO concepts;
- WSMX (Web Services Execution Environment), which defines and provides reference implementation allowing the execution of SWS.

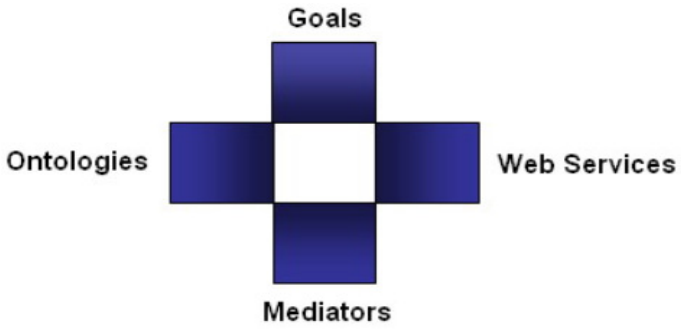

FIGURE 2.1 Four Elements of WSMO

2.3.1 ONTOLOGY

Ontologies provide the formal semantics for the terminology used within all other WSMO components. Ontologies are a key enabling technology for the semantic web. They interweave human understanding of symbols with their machine-process ability.[4]

Ontology first came in view in 1991. The idea was to devise new ways of constructing knowledge-based systems, so that the knowledge base upon which they are based did not have to be constructed from scratch, but by assembling reusable components. Ontology consists of entities, attributes, inter-relationships between entities, domain vocabulary and factual knowledge—all connected in a semantic manner. Concepts are descriptions of things that exist in the domain of the ontology. For example, a banking ontology would probably include concept definitions for bank, account, customer, deposit, loan, and so on. Concepts may contain attributes with names and types. Relations describe interdependencies between multiple concepts. The relation marriedto describes interdependency between a man and a woman. Functions are special relations that result in a single typed value. Ontologies describe the conceptual model for a particular domain; instances are the actual facts described using these concepts. Ontology is often viewed as the key means through which the Semantic Web vision can be realized.

Ontology agrees upon unambiguous vocabulary to be used in reusable and sharable form. Originally introduced by Aristotle, ontology is a formal model about how we perceive a domain of interest and provide a precise, logical account of the intended meaning of terms, data structures and other elements modeling the real world. [1,6] The term ontology, originating from Philosophy as detailed in was adopted by AI researchers to describe formal domain models. The most frequently cited definition is that given by Gruber in 1993 according to which ontology is "an **explicit specification** of a **conceptualization**". In other words, ontology is a domain model (*conceptualization*) which is *explicitly* described (*specified*). Later, in 1997 Borst defined ontology as a "**formal** specification of a **shared** conceptualization". Conceptualization should be expressed in a machine readable format (*formal*). In 1998, Studer et al, merge these two

definitions stating that: “An ontology is a formal, explicit specification of a shared conceptualization.” Ontology provides a common understanding of a particular domain. As domain models, the primary role of ontology is to enhance communication between humans, organizations, and application systems. (establishing a shared vocabulary, explaining the meaning of the shared terms). As formal models, ontology represents knowledge in a computer processable format thus enhancing communication between humans and computer programs or two computer programs. [7, 9]

Ontology provides interoperability among software systems and improve the design and the quality of software systems. Ontologies glue together two essential aspects that help to bring the web to its full potential:

• Ontologies define formal semantics for information, consequently allowing information processing by a computer.

• Ontologies define real-world semantics, which make it possible to link machine processable content with meaning for humans based on consensual terminologies. [6]

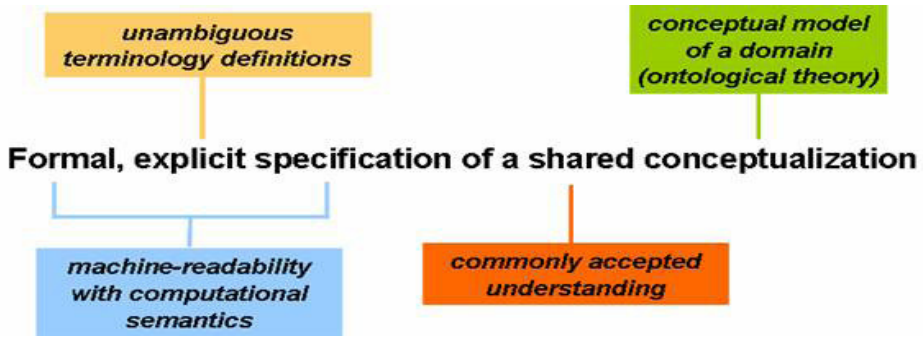

FIGURE 2.2 Ontology Definition

Semantic Web service descriptions rely on two kinds of ontologies. First, a generic Web service ontology specifies the main aspects used to describe Web services regardless of the domain in which they operate. Second, a domain ontology, that has the lowest reusability and are specific to a particular domain. Web service domain ontology contains domain specific knowledge which is used to complete the generic descriptions.

2. 3.1.1 Need of Ontology

The openness of the Web means the volume of published information is growing exponentially resulting in what is commonly termed 'information overload'. To search for relevant data in such a huge amount of collection is becoming very tedious and difficult. Each search engine uses its own secret algorithm when determining, what results to give back and in what order the results should be displayed. It can often be difficult to extract relevant information from the retrieved search results. Sometimes, relevance can only be determined by shifting through the result, one by one. Ontologies help in querying knowledge on the Web by enriching information with descriptions of its meaning. Significantly, these rich descriptions can be interpreted by computer systems, which allow them to interpret the results of Web queries intelligently. Ontologies enables web to share knowledge and provide reasoning support, i.e., verification of formalisms and rules. Checking for constraint violations in knowledge shared through the ontology can be automated. [8, 9]

2. 3.1.2 Use of Ontologies

Ontologies are used in different research areas which include knowledge representation, natural language processing, information retrieval, databases, knowledge management, on line database integration, digital libraries, geographic information systems, visual in formation retrieval or multi agent systems. In e-commerce, ontologies are used to enable communication between buyer and seller. In search engines, ontologies are used to find web pages containing semantically similar words provided by the user. Knowledge about the application domains is very important in order to have successful software projects. This knowledge is usually elicited from domain experts that know about business processes, customer relations, other relevant processes, products and relationships. In this case a powerful ontology for modeling the domain knowledge can be very helpful. In general ontologies are used in the research on data-interchange, data integration, data querying and data-verification. [10]

2.3.2 GOALS

It formally describes the client objective. It states the problem that a client wants to achieve by using web service. A service requester uses Goals to represent the type of service that they are seeking by specifying what capability they would like that service to offer and what public interface they would like it to provide. Service descriptions are intended to provide detailed descriptions of the mechanics of how a service provides its capability and behavior. Goal descriptions describe what capability and behavior the requester would like to find. The Goal is described in terms of ontologies used by the requester. The ability to model both Goals and Web Services provide a distinct conceptual separation between the points of view of service requesters and providers. [1, 11] This allows more flexibility in how service requesters and providers are brought together than is possible with current Web Service technology. Its purpose is to facilitate problem-oriented and dynamic web service usage.

Discovering a web service in UDDI repository and invoking that web service requires the following steps:.

- a suitable service must be located in a UDDI repository. UDDI supports this step with keyword search.
- If a service is located, its textual description is checked to see if it fits the requirements. However, as service descriptions provided in UDDI are informal, the requester must assume that their understanding is the same as that intended by the service provider.
-If the requester is satisfied with the Web Service, the associated WSDL document provides the syntactic description of what messages the service accepts and what transport protocol to use when interacting with the service.
-To make an invocation of the Web Service, the requester may have to adjust their data to fit the service description.

The above stated discovery procedure would require the interaction between service requester and service provider to be tightly coupled together. If the requester wants to use another banking service later, they will have to repeat the entire process of finding and binding to a suitable service again.

Describing both Goals and Web Services separately using the Web Service Modeling Ontology shifts the responsibility of matching service requests to service descriptions from the requester to Semantic Execution Environments, such as WSMX. [1]

2. 3.2.1 Goal specification

Goal specifications should be kept separate from actual web service descriptions because the same web service can serve different goals and obviously different (competing) web services can serve the same goal. For example, Amazon could be used to buy a book, however, in the same way it can be used as an information broker on bibliographic information about books. Conversely, different book stores may subscribe to the same goal. At least one ontology is always imported by a goal specification to further define the terms that are used to describe it. A goal specification consists of two elements:

• **Pre-conditions** specify what a web service expects for enabling it to provide its service. For example, asking for the nearest restaurant only makes sense if requestor can tell the web service about his geographical location i.e. where he is standing presently. Asking for the *best* hotel in a city requires that user's requirements should be specifically mentioned.

• **Post-conditions** specify what a web service returns in response to its input. In response to geographical information it may return the closest restaurant or it may return the best hotel according to the preferences of a client. [12]

2.3.3 WEB SERVICE

The Web Services represent the functional part: each Web Service offers functionality which has to be semantically described, to enable its (semi-) automatic use. The term "Web Service" is usually understood as a composition of three major elements:

-Interface descriptions captured by WSDL documents

-The communication protocol, SOAP using XML to exchange messages

-UDDI repositories allowing potential users to find services that are offered by providers.

Web services are applications that can communicate with other applications over a network by using a set of standardized protocols. Web services provide a platform for information transmission, which is independent of programming language, software

platform and hardware. Web services reflect a new, service-oriented approach to programming. This approach is based on the idea of building applications by discovering and implementing network services or by invoking available applications to accomplish some task. This approach is independent of specific programming languages or operating systems. Once a Web service is deployed, other applications (and other Web services) can discover and invoke the deployed service. Web services deliver interoperability; the ability for components created in different programming languages to work together as if they were created using the same language. Web services rely on existing transport technologies, such as HTTP, and standard data encoding techniques, such as Extensible Markup Language (XML), for invoking the implementation. A typical Web services scenario is a business application requesting a service from a given URL using Simple Object Access Protocol (SOAP) over a HyperText Transport Protocol (HTTP) transport. The service receives the request, processes it, and returns a response. Examples of a simple Web service include weather reports and stock quotes. The method call is synchronous, that is, it waits until the result is available.

2. 3.3.1 Web Services Architecture

The Web services architecture includes three roles:

- **Service provider:** Web service providers create components, then publish them to a repository. Web service providers can also unpublish the components (remove them from the repository) when they are no longer needed.
- **Service broker:** Web service brokers categorize Web services as they are published, and search for them as service requests are received. Web brokers are roughly analogous to Internet search engines, except that they locate components instead of Web pages.
- **Service requester:** Web service requesters look up, or locate and invoke components as services. They act as the client for published Web services.

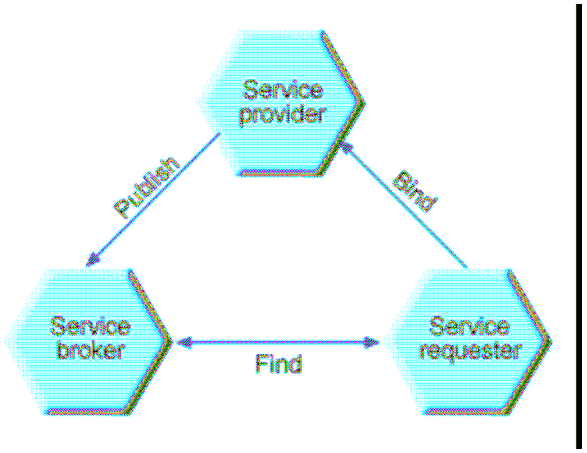

FIGURE 2.3 Client and Server Interaction To Provide Web Services

2. 3.3.2 Web services operations

Web services reduce programming complexity because application designers do not have to worry about implementing the services they invoke. Interactions in Web services are bound dynamically at runtime. A service requester describes the features of the required service and uses the service broker to find an appropriate service. A Web services component has following life cycle:

- **Creation:** The Web services provider creates the service component by defining its interfaces and invocation methods.
- **Publication:** The Web services provider publishes the service component to a repository. The Web services broker categorizes the new Web service within its listing.
- **Location:** The Web services requester looks up, or locates, a Web service component through the service broker.
- **Invocation:** Once the service requester locates the service component, it invokes and implements it.
- **Unpublication:** When the Web service provider decides that a Web service should no longer be available, it removes, or unpublishes, the Web service from the repository. The service broker likewise removes the service component from its listing.

In the WSMO context, a Web Service is a formal description required to enable the automatic processing of Web Services. WSMO focuses on the external Interface of the Web Service, while its internal implementation remains out of the scope of WSMO. The Web Service description in WSMO provides rich descriptions enabling not only humans, but also software entities to understand the capabilities and interfaces of the service. Such an unambiguous description of a web service with well-defined semantics can be processed and interpreted by software agents without human intervention. This enables the automation of the tasks involved in the web service usage process such as discovery, selection, mediation, composition, execution and monitoring. [13]

2. 3.3.3 Functional Aspect

The functional aspects of a web service are described by the Capability of the service. The Capability makes automated Web Services discovery possible. Web Service Capabilities are defined by four notions:

- **Preconditions** - describes what a web service expects in order to be able to provide its service. They define conditions over the input. For example, for the e-banking web service, in which the comparison of the mortgage interest rates being offered by banks online are made, these can be inputs, which have to be provided by a client (i) an amount of money, which client would like to borrow and (ii) repayment period for a requested mortgage.
- **Assumptions** - conditions on the world that have to hold before execution, for example in the e-banking case, the fact that a client is coming from a member country of European Union would be an assumption.
- **Post conditions** - describes the result of the Web Service in relation to the input, and conditions on it.
- **Effects** - conditions on the world that hold after service execution. For example, as a result of Web Service execution, money may be transferred to client account.

2. 3.3.4 Behavioral Aspects

The behavioral aspects of the web service are addressed by the *Interface* of the service. It contains:

- **Choreography***:* It expresses the interface for consumption. It specifies how to interact with the service to consume its functionality.
- **Orchestration***:* It defines how functionality can be achieved by aggregating other Web Services.

2. 3.4 MEDIATORS

Exchanging data and share functionality has been a great challenge both from the technological and efficiency point of view. There exist mismatches on structural, semantic, conceptual, level between different components that shall interoperate especially in distributed & open environments like the Internet. For example, business entities having completely independent applications willing to interact with each other, can not communicate because of differences in ways of representing and structuring data. This drives the need for mediators, third-party systems able to deal with the potential mismatches that may appear both on the data and behavioral level between the interacting parties. Mediators are components that resolve mismatches. The techniques used in developing mediators have to be dynamic and scalable. Mediators should be flexible systems and easy to extend, assuring loose coupling between various business entities. Mediators offer intelligent mechanisms that resolve mismatches independent of content. [10]. WSMO provides four classes of mediators able to cope with the heterogeneity problems that might occur between ontologies, web services and goals:

- **ooMediators** describe the class of mediators able to solve the heterogeneity problems between ontologies. Since ontologies are developed independently by various business entities, one can find ontologies describing the same domain in different terms and, without mediators, applications using these kinds of ontologies would not be able to exchange data. ooMediator points to a concrete mediation solution, able to actually solve the heterogeneity problems between the specified source and target ontologies.

- **ggMediators** are used for coping with the differences and for exploiting the similarities that may exist between different goals. Any ggMediator may use the services of ooMediators, in case the goals, between which it mediates, are expressed using different ontologies.
- **wgMediators** are the class of mediators that address the heterogeneity problems between a goal and a Web service at two different levels:
  Functionality : to check whether the Web service completely satisfy the goal
  Communication *:* to find out how the two partners can communicate.
- **wwMediators** are the most complex class of mediators in WSMO, addressing the heterogeneity problems between different Web services. These problems may occur when a Web service is invoking one or many other Web services in order to achieve certain functionality, and implies three levels of mediation: functionality, communication and cooperation.

Levels of Mediation in WSMO:

– **Representation:** heterogeneous Languages & Protocols

– **Data Level:** heterogeneous Data Sources

– **Functional Level:** heterogeneous Functionalities

– **Process Level:** heterogeneous Communication Processes

2.4 Web Service Execution Environment (WSMX)

Creating ontologies and semantic descriptions for Web Services is only useful if these descriptions can ultimately be applied. An execution environment for Semantic Web Services is the infrastructure required to enable automated discover, mediation, selection, invocation, and interoperation of these services. WSMX is a reference implementation of WSMO and takes the full conceptual model of WSMO into consideration. [14] The goal of research in WSMX is to provide an architecture for Semantic Web Service based systems. WSMX is based on the conceptual model provided by Web Service Modeling Ontology, which describes various aspects related to Semantic Web Services. WSMX is also a reference implementation for WSMO, providing the proof of its applicability and usefulness as well as being a vehicle for driving new projects and partnerships. [15]

WSMX is an execution environment for finding and using Semantic Web Services that are described using WSMO. The user provides WSMX with a WSMO Goal that formally describes what they would like to achieve. This enables the automation of previously human intensive tasks when building applications with a Service Oriented Architecture. Various features of WSMX are:

Discovery: Determine usable services for a request. WSMX uses the Discovery component to find Web Services, which have semantic descriptions registered with WSMX that can fulfill this Goal.

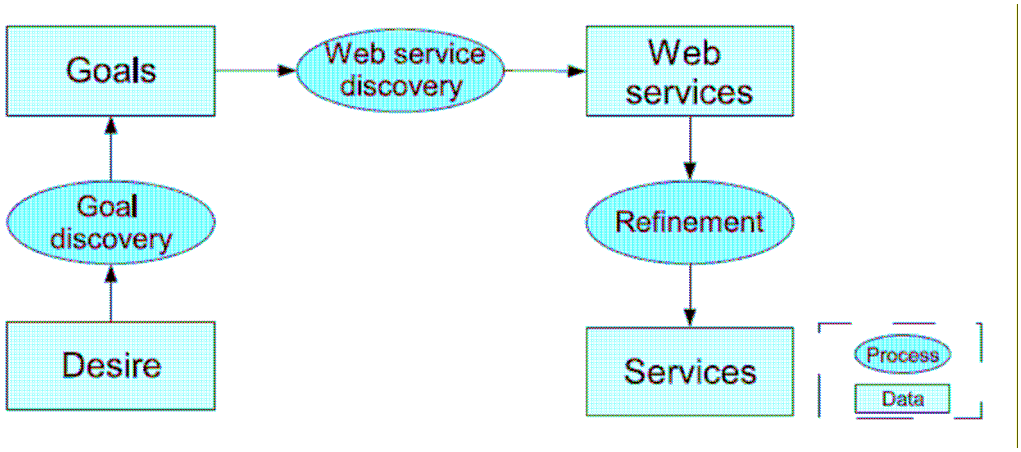

FIGURE 2.4 Discovery of Desired Goal

Composition: WSMX combines the related services to achieve a goal.

Ranking and Selection: Choose appropriate service for the job

Mediation: WSMX solves the mismatches to enable interoperability. One of the most difficult obstacles Web Services have to overcome in the attempt to exploit the true potential of the World Wide Web is heterogeneity. During the discovery process the users Goal and the Web Services description may use different ontologies. If this occurs Data Mediation is needed to resolve heterogeneity issues. Data Mediation in WSMX is a semi automatic process that requires a domain expert to create mappings between two ontologies that have an overlap in the domain that they describe. Once these mappings have been registered with WSMX the runtime data Mediation component can perform automatic mediation between the two ontologies. [3, 14] The Process Mediation

component in WSMX is responsible for resolving mismatches between the Choreographies of the user and Web Service.

Invocation: WSMX executes the entry points on the service. Once this mediation has occurred and a given service has been chosen that can fulfill the users goal, WSMX can begin the process of invoking the service.

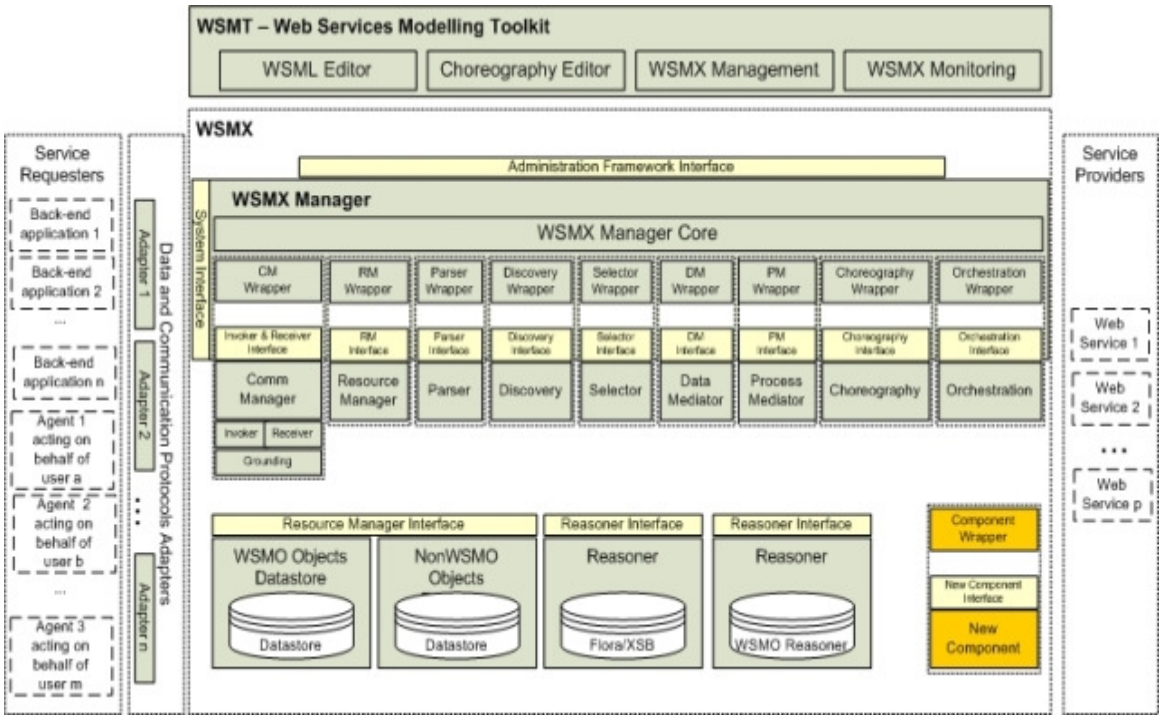

FIGURE 2.5 WSMX Architecture [15]

2.5 Web Services Modeling Toolkit (WSMT)

The Web Services Modeling Toolkit (WSMT) is a framework for the rapid creation and deployment of homogeneous tools for Semantic Web Services. It aims to support the developer through the Software Development Cycle (SDC) of Semantic Web Services by improving Developer Productivity, aid in adoption of WSMO, WSML, SEE .[17] It provides high quality Eclipse based tools. The WSMT was designed to be the front-end of the WSMX system and provides a number of tools to users.

The WSMT is wrapped in a full installation system, which allows the end user to choose the tools that are provided during the installation process. An initial set of tools for the

WSMT includes: a WSML Editor for editing WSML documents and publishing them to WSMO repositories, a WSMX Mediation tool for creating mappings between Ontologies.[18].

Chapter 3

LITERATURE REVIEW

Recent progress in the field of Web services makes it practically possible to publish, locate, and invoke applications across the Web. The ability of efficient selection and integration of inter-organizational services on the Web at runtime becomes an important issue to the Web service usage enhancement.

The Web Service technology, allows uniform access via Web standards to software components residing on various platforms and written in different programming languages. As a result, software components providing a variety of functionalities are now accessible via the Web. Web service is seen as an application accessible to other applications over the Web. Web services are open, which means that services have published interfaces that can be invoked by message passing standards. [24]

The goal of the Semantic Web is to solve the current limitations of the Web by augmenting Web information with a formal (i.e., machine processable) representation of its meaning. A direct benefit of this machine processable semantics would be the enhance ment and automation of several information management tasks, such as search or data integration. The Semantic Web develops techniques for augmenting existing Web data with logics based formal descriptions of their meaning. [23]

A major limitation of the Web services technology is that finding and composing services still requires manual effort. If no single Web service can satisfy the functionality required by the user, there should be a possibility to combine existing services together in order to fulfill the request. The Web services are usually developed by different organizations that use different conceptual models for presenting services characteristics. This requires utilization of relevant semantic information for matching and composition of Web services. This becomes a serious burden with the increasing number of Web services. To address this problem, semantic Web researchers proposed to augment Web services with a semantic description of their functionality in order to facilitate their discovery and

integration. This technology, combining Web services and Semantic Web techniques, is referred to as Semantic Web Services. [2]

One of the crucial tasks to be performed towards the realization of the vision of the Semantic Web is the encoding of human knowledge in special structures. These special structures are encoded as ontologies. Ontologies are formal models about how we perceive a domain of interest and provide a precise, logical account of the intended meaning of terms, data structures and other elements modeling the real world. Ontologies can help in the representation of the content of a web page in a formal manner, so as to be suitable for use by an automated computer agent. Ontologies are core concept of the Semantic Web that allows the representation of data in a machine processable way. Ontologies are often large and complex structures, whose development and maintenance give rise to several sturdy and interesting research problems. [7]

WSMO is a complete conceptual model for Semantic Web Services and related aspects. WSMO is derived from and based on the Web Service Modeling Framework (WSMF). WSMO aims to solve the integration problem. WSMO identifies four main elements: Web Services, Goals, Ontologies, and Mediators. Ontologies provide the terminology used by other WSMO elements, Web service descriptions describe the functional and behavioral aspects of a Web service, goals represent user desires, and mediators aim at automatically handling interoperability problems between different WSMO elements. WSMO allows applications to be as much independent as possible. [5]

The Web Service Modeling Toolkit focuses on providing a graphically based editing environment with special support for connecting to execution environments. The toolkit combines a number of separate editors and views. Its main focus is on graphical visualization and support for modeling, executing and monitoring Web services. The graphical editor of WSMT uses a semiautomatic approach in which a component first suggests a mapping which the user must later refine. Once these mappings have been created, they can be used to transform instances of the source ontology into instances of the target ontology. [17]

The execution environment of Semantic Web services is a framework which serve as an architecture to integrate the tasks involved in the use of Semantic Web services from goal specification to execution. The amount of human interaction required in the process of finding and using Web services is a serious limitation on their use. Since WSMO allows Web services to be described semantically, a large portion of this process can be automated. The Web Service Execution Environment (WSMX) is a reference implementation of WSMO and is an execution environment within which semantic descriptions of user's goal and provider's Web service can be used to discover, compose, mediate, select and invoke Web services that match the end user's requirements. WSMX does not define services in terms of their design or implementation details, but only in terms of expected functionality and their externally visible behavior. WSMX can simply discover and return the services found to the user, allowing the requester itself to perform selection. [14]

In WSMX, much of the process of finding and using Web services can be automated. Automatic discovery offers a great advancement over the current situation, allowing users to find services that can fulfill their requirements without spending large amounts of resources. Mediation and invocation also reduce the effort of manually interpreting interfaces, inputs, and outputs of Web services and switching data formats to fit the formats required by those Web services. [20]

Chapter 4

PROBLEM STATEMENT

4.1 PROBLEM DEFINITION

During literature review it was analyzed that semantic web as an extension to current web services will lead to automated and dynamic computations over web. Semantic web technology relies on ontology as data representation form for formalization and conceptualization the data. Various discovery and heterogeneity problems arise as ontologies are developed by different domain experts in isolation, using different representation forms and data structures. There is a requirement of automated discovery and invocation of web services. The problem is how to integrate applications that are using different conceptualizations of the same domain.

To overcome heterogeneity and interoperability in semantically enriched environment data mediation is required. Various frameworks have been provided by research community for designing, developing, deploying, mapping and merging of Semantic web Services. Major frameworks considered so far include OWL-S, WSMO and WSDL-S. Our main emphasis is on WSMF. WSMF is a framework, which serves as the basis of WSMO conceptualization. It provides various features in a toolkit, WSMT. Ontologies are mapped; web services are discovered and deployed. The mappings can be done manually, but this has proven to be a very laborious, error-prone and time consuming process. The mapping creation process cannot be performed fully automatically and as a consequence, the domain expert has to provide inputs at various stages of this process. We have tried to work on the mapping tools offered by WSMO graphical interfaces to assist the user in the mapping creation and to reduce human effort to simple choices and validations.

4.2 METHODOLOGY

The step-by-step methodology to be followed is given below:

- Study various inbuilt features of WSMF including WSMO, WSMX, etc.
- Review the inbuilt ontologies in various views provided by WSMT.
- Import the user developed ontologies.
- Query both inbuilt and imported ontologies.
- Mapping of imported ontologies and generating inferences in WSMT.
- Adding web services in semantic execution environment by installing Tomcat and generating results through WSMX.

Chapter 5

IMPLEMENTATION

5.1 BROWSING AND EDITING ONTOLOGIES USING WSMT

5.1 1 Ontologies views

WSMT provides three types of editors to view the ontology.

(i)WSML TEXT EDITOR: The WSML Text Editor shows the contents of a WSML document, in the WSML human readable syntax to the user with syntax highlighting, content assistance and many other features that help advanced users who are familiar with WSML to create and manage their WSML documents more efficiently.

(ii) WSML VISUALIZER: The graphical view of any ontology can be seen in its WSML VISUALIZER view. The WSML visualizer provides editing and browsing support in same interface. It provides different notations for concepts, instance, web service, capability etc. This feature provides semantic highlighting in graphical view. Graph based representations give a better feel for the complexities of a semantic description

(iii) WSML FORM BASED EDITOR: It represents ontology in a form that consists of text fields, combo boxes and tables. For creating this view descriptions are broken up into tabs and relevant information from textual content is shown in the respective tab.

WSMT ensures consistency. Changes made in one view are automatically reflected in two other views. Any element of the ontology can be added, deleted or modified in any of the view.

Figure 5.1, 5.2, 5.3 shows the text editor view, visualize view and form based view respectively of Simpsons ontology. Simpsons ontology is an inbuilt ontology in WSMT.

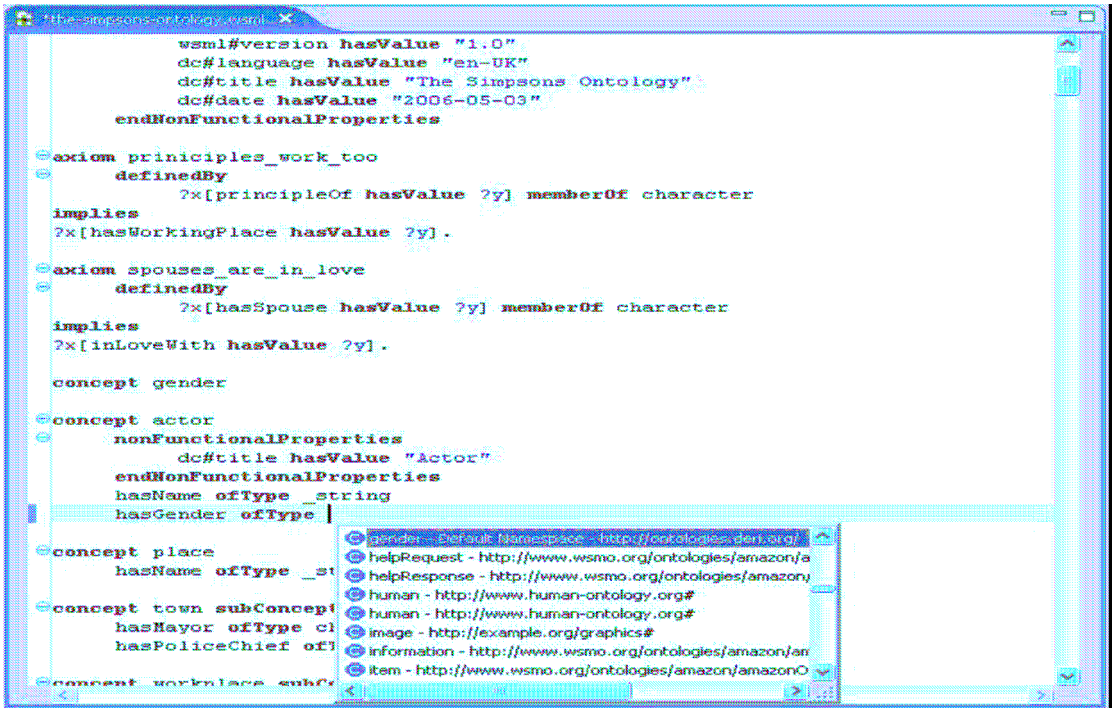

Figure 5.1. WSML Text Editor View

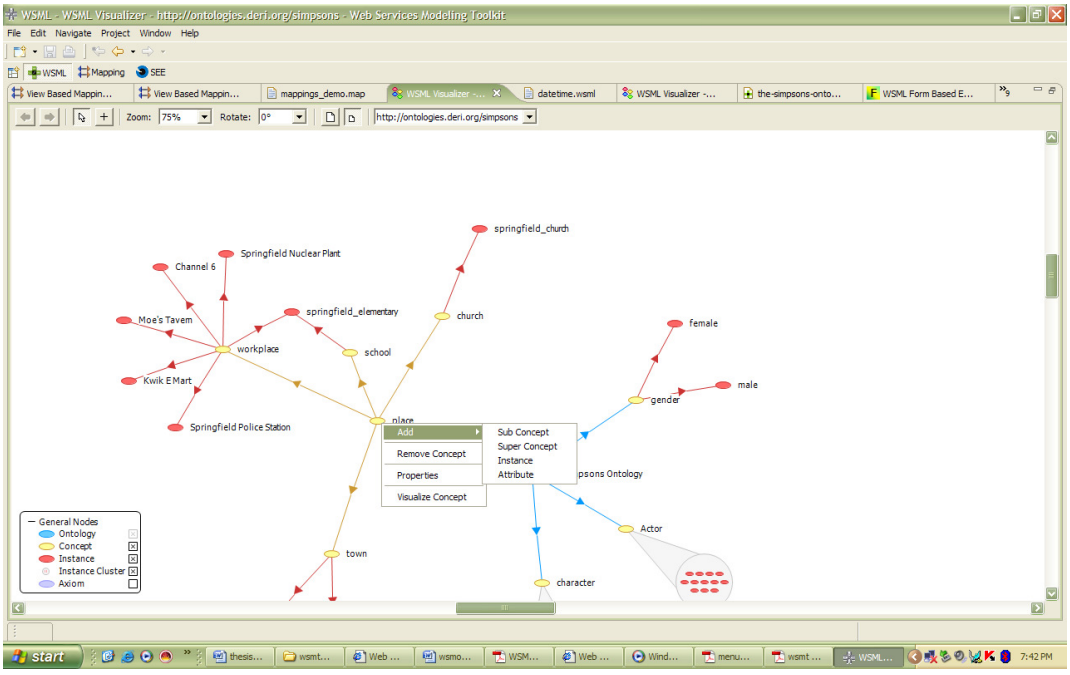

Figure 5.2 Graphical View of Ontology in WSMT

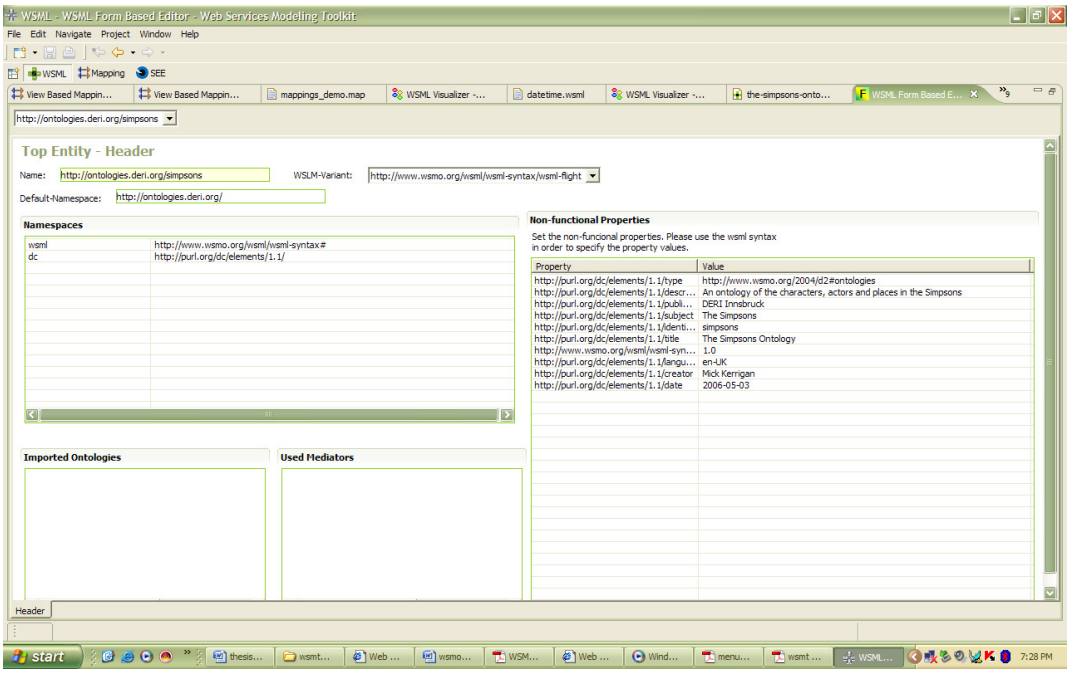

Figure 5.3 Form Based Editor

5.1.2 Importing Ontologies

WSMT is provided with some inbuilt ontologies, there mappings, mapping results. User can import its own ontology in editor and can perform various functions on these imported ones too. Importing feature is provided with semantic execution environment perspective. SEE provides interface with WSMX server. WSMX server on port 8080/8081 provides complete namespaces for various ontologies, web services, goals. The ontologies, web services, goals corresponding to these namespaces can be stored in WSMT workspace directly. On right clicking any of the namespace, it gives the option to store the file corresponding to that namespace in its workspace. Its snapshot is provided with section 5.3. Files can be imported to WSMT interface and exported from here to any other location in WSML perspective too. On right clicking any file in WSML perspective, it gives options for import and export. On clicking import it asks for source file and on clicking export it asks for export destination.

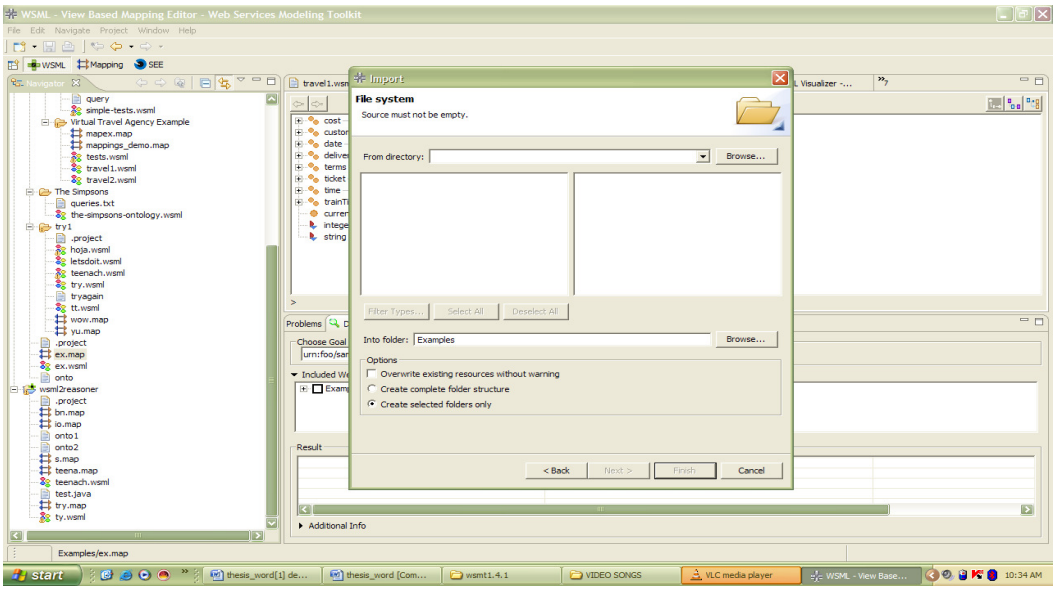

FIGURE 5.4 Importing File in WSML Perspective

5.1.3 Querying Ontology (REASONER)

Queries can be made on the ontologies. A data reasoner is provided for querying. It supports basic SPRQL query structure and returns result in a tabular form. Figure 5.5 shows the WSML reasoner view. A query is made on ontology and the result is displayed in tabular form.

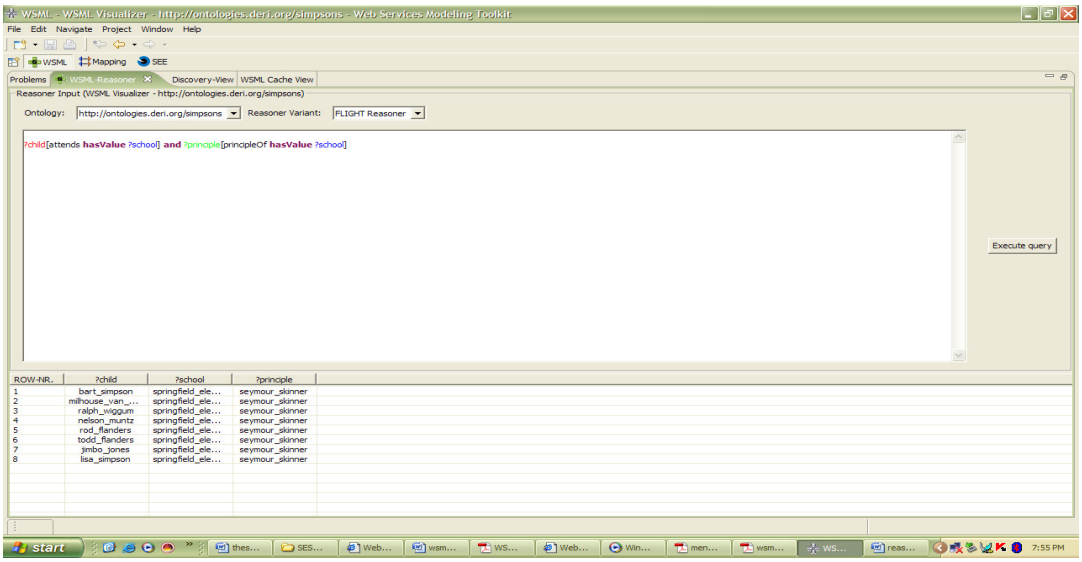

FIGURE 5.5 WSML Reasoner Result after Querying

5.2 MAPPING ONTOLOGIES

The mappings have the role of expressing the links between the source and the target ontology. Two ontologies can be mapped with this toolkit. A mapping file is created, in which one is taken as source and other is taken as target ontology. Source concepts and attributes are mapped with concepts and attributes of other target ontology. Both inbuilt and imported ontologies can be used as source and target ontology. Mapping unit takes inferences from the user and stores them in a repository. It leads to overcome heterogeneity problem among ontologies. For generating valid results, care should be taken while selecting source, target ontology and while providing inferences. Source and target ontologies should satisfy at least one common goal.

While providing inferences to the WSML mapping unit, the unit also guides us to valid mapping elements. Figure 5.6 shows the creation of a mapping file. For creating a new file it first asks for source and target ontology. Source instances are mapped to target instances.

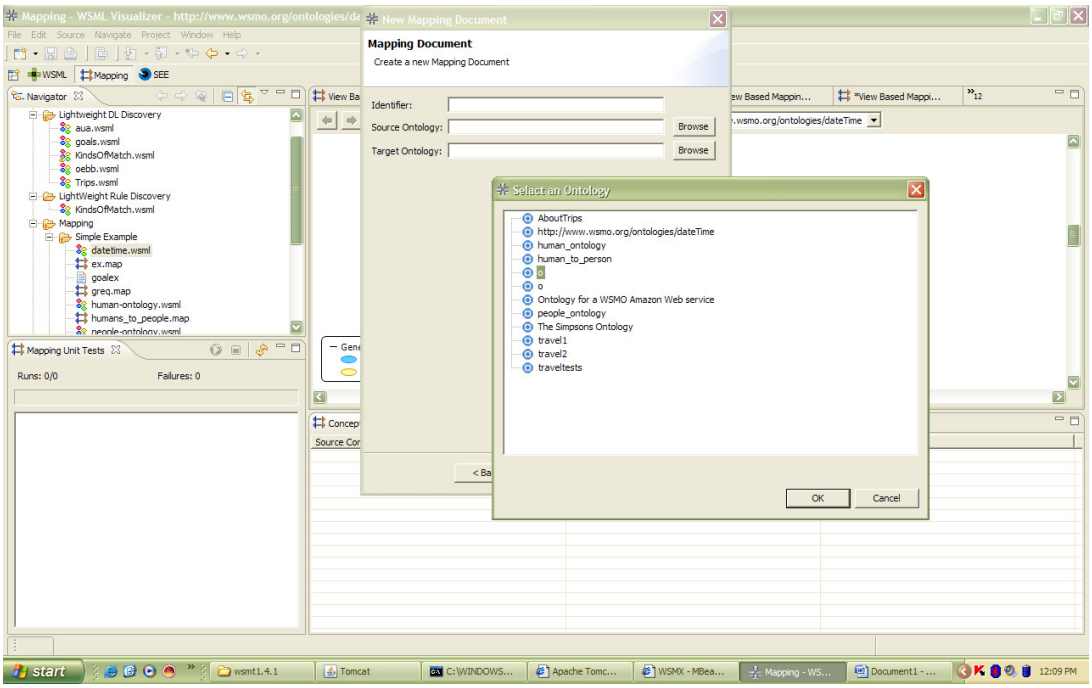

FIGURE 5.6 Creating Mapping File

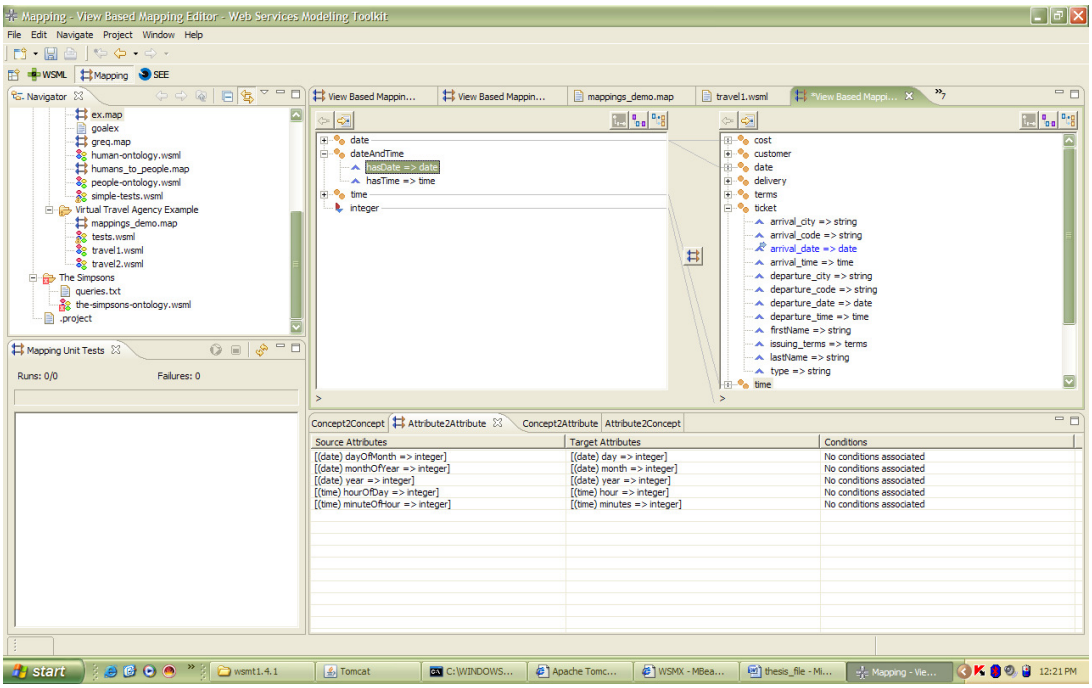

FIGURE 5.7 Generation of Inferences

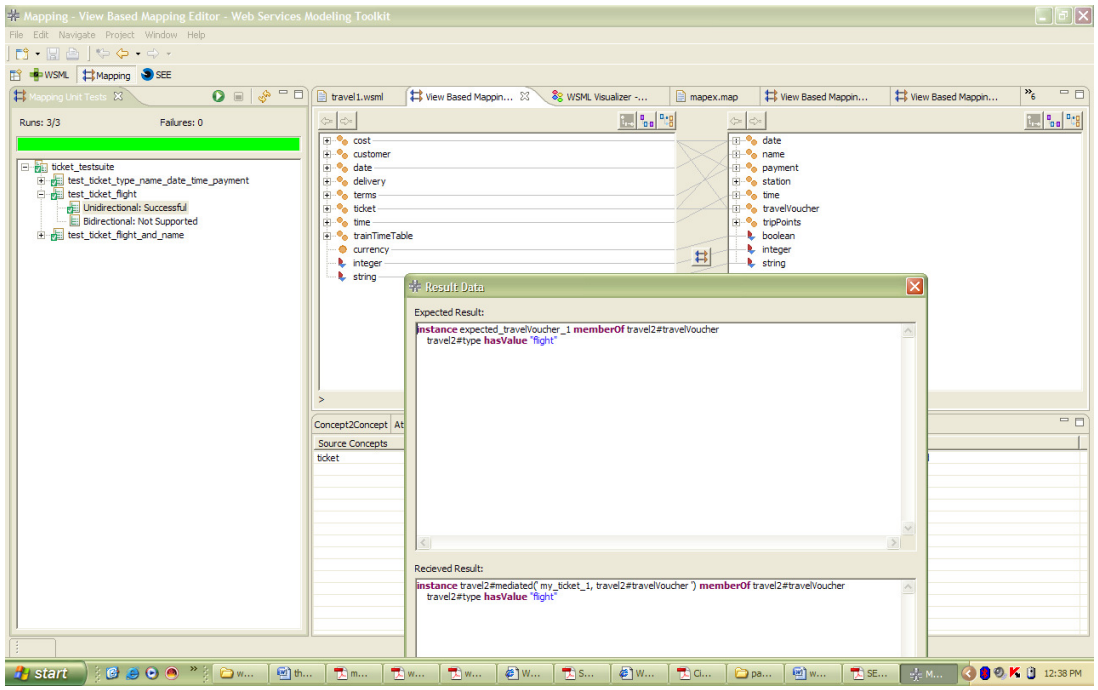

FIGURE 5.8 Mapping Result

It provides mapping information as mapping between attribute to attribute, concept to concept, concept to attribute and attribute to concept, where first elements corresponds to source element and later one corresponds to target element. Figure 5.7 shows a mapped file. The ontology date.wsml is taken as source and travel.wsml is taken as target. It executes the mapping inferences and provides result as expected and received instances. Figure 5.8 shows the execution result.

The mapping process basically requires three types of actions from the domain expert:

- **Browse the ontologies**: The domain expert has to discover the ontology elements of interest for this particular task.
- **Identify the similarities**: This step involves the identification of semantic relationships between the entities that are of interest in the two ontologies. For this the user can make use of the suggestions offered by a set of lexical and structural algorithms for determining semantic relationships.
- **Create the mappings**: The last step involves capturing these semantic relationships by mappings. This means that the domain expert has to take the correct actions in order to properly catch the semantic relationships in the mapping language statements or may use predefined mapping patterns.

5.3 SEMANTIC EXECUTION ENVIRONMENT VIEW

In order for a SEE to correctly function the necessary ontologies, Goals, Web Services and Mediators should be available to it. Automated tools for interfacing with the Web Services exposed by a SEE enable deploying descriptions to a SEE or retrieving them in order perform maintenance. Figure 5.9 shows the SEE perspective of WSMT. It works with tomcat as wsmx server. WSMX server is added at 8081. It provides WSMO elements at url http://localhost:8080. Ontology and goals can be added to workspace through this perspective directly.

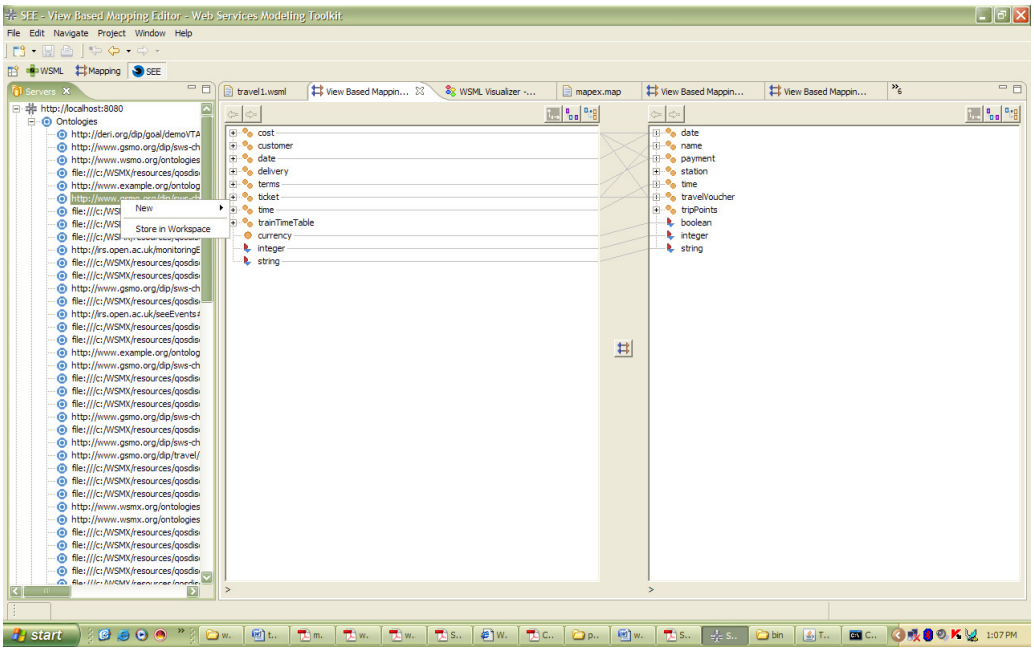

FIGURE 5.9 SEE View

5.4 WSMX

Web console of WSMX is opened with URL http://localhost:8081.WSMX page provides four views.

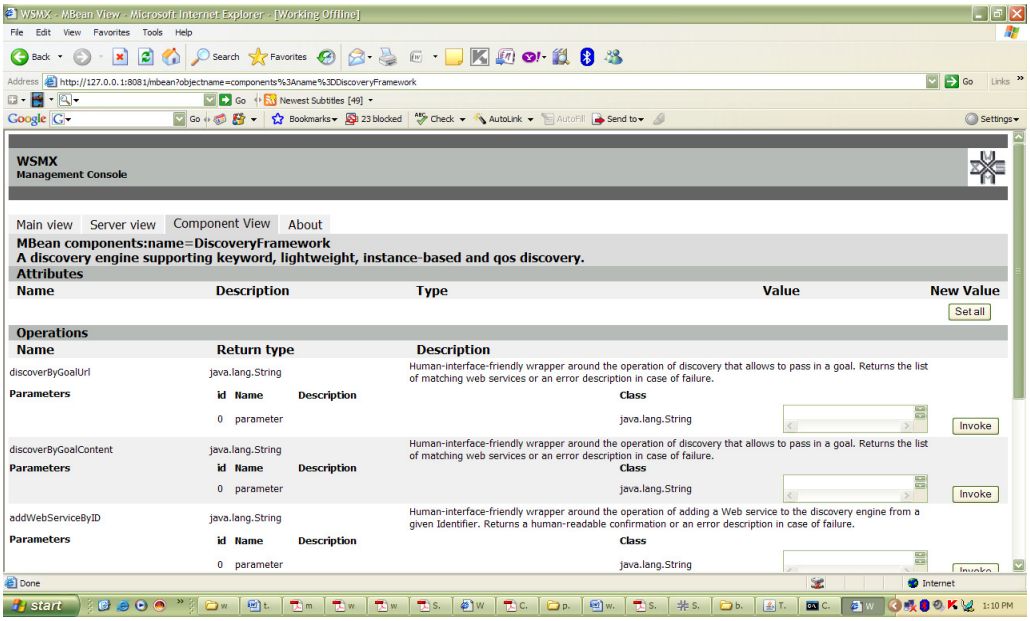

FIGURE 5.10 WSMX Page

The default view is Main view. This page shows the general system information like name of host computer, network IP address, space address, information about other federated WSMX peers, startup time and WSMX system uptime duration. Service view provides information about the Class loaders, management interface of the Mbean implementation (listing all the attributes, constructor, destructor and methods with corresponding input and return types). Component view provides the information about components deployed on the WSMX. Here web services, goals and ontologies can be deployed on server and can be discovered from server. Discovery can be done on the basis of content or url.

Discovery feature is embedded with WSMT also. It is provided as discovery view. Figure 5.11 shows the discovery view of WSMT. In figure 5.11 'TrainTrip' is chosen as goal and DL as formalism. On pressing Discover button, only the Web Service that is related to train travel (i.e. oeeb) is shown in result table, as the Goal TrainTrip asks for all the services that are related to travel by train only.

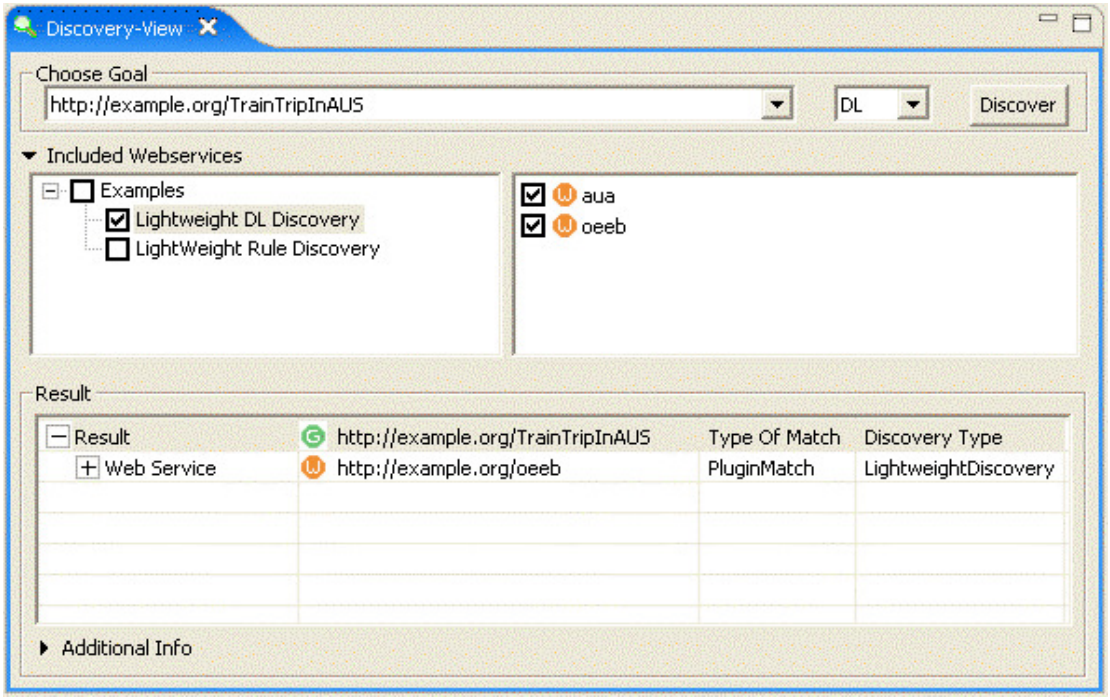

FIGURE 5.11 Discovery View in WSMT

Chapter 6

CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

Web Services have become another milestone towards providing interoperability among distributed and independent software systems. But one major problem has remained unresolved: **interoperability**, which has to be addressed to enable dynamic collaboration of independent software entities on the Internet. Data that is exchanged between Web servers and Web browsers remains solely dedicated for human consumption, and cannot be readily processed by automatic software agents. Adding semantics to the existing Web Services technologies is a fundamental requirement if we want to deliver workable integration solutions for the next Web generation.

Semantic Web Services are one of the most promising research directions to improve the integration of applications within and across enterprise boundaries. In this context, various frameworks are modeled, WSMO is one of them. WSMO has the aim of providing the conceptual and technical means to realize Semantic Web Services, improving the cost-effectiveness, scalability and robustness of current solutions. Web Service Modeling Ontology is a meta-ontology for describing several aspects related to Semantic Web Services. WSMO and its related specifications and technologies are principal candidates to become the backbone on the next Web generation, enabling software entities to dynamically interoperate over the Internet.

In this thesis, we worked on a modeling framework called Web Service Modeling Framework (WSMF). Its main elements are: ontologies, goal descriptions, elementary and complex web services, and mediators. The aim of WSMF is to enable fully flexible and scalable e-commerce based on web services. This approach is based on the semantic description of the Web Services and of their behavior, as well as on how a requestor that wants to interact with a Web Service expresses the expected behavior of the Web Service.

The Web Services Modeling Toolkit provides a useful framework for the rapid development of tools for Semantic Web Services. It provides a graphical view of WSMO's main elements, relaxing users from dealing with language code to understand various elements. It provides a semi-automatic approach to ontology-to-ontology mediation problem. The mediation process consists of three main steps: mapping generation, mapping rules creation and execution. The mapping creation process cannot be performed fully automatically and as a consequence, the domain expert has to provide inputs at various stages of this process. Of course, the mappings could be done entirely manually, but this has proven to be a very laborious, error-prone and time consuming process. Considering this, the mapping tools offer graphical interfaces in order to assist the user in the mapping creation and to reduce human effort to simple choices and validations. The mapping tool is integrated as a plug-in in the Web Service Modeling Toolkit. The editor can be used by the domain expert to operate on the ontologies, they are mapping and the framework offers basic synchronization functions like reloading the ontologies into the mapping tool or invalidating mapping rules affected by changes in the ontologies.

6.2 Future Scope of Work

The given work can be extended further.

1. Future work to mediation includes a fully automated mapping generator.
2. The implementation of the concept of run time data mediators, that leads to a fully automated discovery manager.
3. Query generator on mapped files.
4. Addition of more graphical components to which individual tools can contribute; for example an application toolbar and preferences dialog.

REFERENCES

[1]. Jorge Cardoso, Amit P. Sheth, ”Semantic Web Services, Processes and Applications”.

[2]. Michael Stollberg, Matt Moran, John Domingue, and Liliana Cabral, “Semantic Web Services Tutorial”, 5th International Conference on Web Engineering (ICWE 2005) Sydney, Australia, 2005 July 25.

[3].Leopold Franzens, ”SESA Tutorial”.

[4]. Natalya F. Noy and Deborah L. McGuinness Stanford University, Stanford, CA, 94305, “Ontology Development 101: A Guide to Creating Your First Ontology”.

[5]. D. Fensel, C. Bussler, Vrije Universiteit Amsterdam and Oracle Corporation” The Web Service Modeling Framework WSMF”.

[6]. Marta Sabou, “Building Web Service Ontologies.”

[7]. Giorgos Flouris, “On Belief Change and Ontology Evolution.”

[8]. Jorge Cardoso, University of Madeira, Portugal, “Semantic Web Services: Theory, Tools, and Applications.”

[9]. Asunción Gómez Pérez, “ A survey on ontology tools”, 31st May, 2002.

[10]. M. A. M. Nieto (2003). “An Overview of Ontologies”. Tech. rep., Center for Research in Information and Automation Technologies Interactive and Cooperative Technologies Lab Universidad De Las Americas Puebla.

[11]. Dumitru Roman, Holger Lausen, “Web Services Modeling Ontology (WSMO)**”,** SWWS consortium and Project.

[12].John Domingue, Dumitru Roman, and Michael Stollberg**, “**Web Service Modeling Ontology (WSMO) - An Ontology for Semantic Web Services”, W3C Workshop on Frameworks for Semantics in Web Services, June 9-10, 2005, Innsbruck, Austria.

[13].Michael Stollberg and Armin Haller DERI – Digital Enterprise Research Institute, "Semantic Web Services Tutorial", 2005 July 11.

[14].Adrian Mocan, Emilia Cimpian, "D13.3v0.2 WSMX Data Mediation WSMX Working Draft 11 October 2005" URL http://www.wsmo.org/TR/d13/d13.3/v0.2/

[15]."Web Service Execution Environment (WSMX)" URL http://www.w3.org/Submission/WSMX/

[16]."Web Service Execution Environment", url http://www.wsmx.org

[17].Mick Kerrigan,seminar on "The Web Service Modeling Toolkit (WSMT)", Asian Autumn School on the Semantic Web 2007

[18].Mick Kerrigan, "The WSML Editor Plug-in to the Web Services Modeling Toolkit", Digital Enterprise Research Institute (DERI), National University of Ireland, Galway, Ireland.

[19].H. Lausen, J. de Bruijn, A. Polleres, and D. Fensel: WSML - a Language Framework for Semantic Web Services. W3C Rules Workshop. In Proceedings of the W3C Workshop on Rule Languages for Interoperability, Washington DC, USA, April 2005. Position Paper: http://www.w3.org/2004/12/rules-ws/paper/44.

[20].Emilia Cimpian, Adrian Mocan "WSMX Process Mediation Based on Choreographies", DERI, National University of Ireland, Galway, Ireland.

[21]. Emilia Cimpian, Mick Kerrigan, "WSMX Process Mediation", DERI, National University of Ireland, Galway, Ireland.

[22]."D16.1v0.21 The Web Service Modeling Language WSML" URL http://www.wsmo.org/TR/d16/d16.1/v0.21/

[23]. Dieter Fensel, Holger Lausen, Axel Polleres Jos de Bruijn, Michael Stollberg, Dumitru Roman, John Domingue, " Enabling Semantic Web Services The Web Service Modeling Ontology"

[24]. Grigoris Antoniou, Frank van Harmelen, "A Semantic Web primer".

[25]. Peter Mika, “Social Networks and the SemanticWeb”

Appendix A

GOAL TRAIN-TRIP.WSML

```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-dl"
namespace { _"http://example.org/", trip _"http://trip.example.org/" }

goal Trip
        importsOntology {trip#AboutTrips ,_"urn:foo/o"}
        capability mygoalTripCap
        sharedVariables ?x
        postcondition mygoalPost
   definedBy
                ?x memberOf trip#Trip.

goal TrainTrip
        importsOntology {trip#AboutTrips ,_"urn:foo/o"}
        capability mygoalTrainCap sharedVariables ?x
        postcondition mygoalTrainPost
   definedBy
                ?x memberOf trip#TrainTrip.

goal TrainTripIBK2VIE
        importsOntology {trip#AboutTrips ,_"urn:foo/o"}
        capability TrainTripIBK2VIECap sharedVariables ?x
        postcondition TrainTripIBK2VIEPost
        definedBy
       ?x memberOf trip#TrainTrip
  and forall ?from ( (?x[trip#from hasValue ?from]
        implies ?from memberOf trip#PlaceInInnsbruck) )
  and forall ?to ( (?x[trip#to hasValue ?to]
        implies ?to memberOf trip#PlaceInVienna) ).
```

LIST OF PUBLICATIONS

• Deepti Gupta and Shalini Batra, "*Ontological Engineering*" **Published** in National Conference on Communications & Networking (NCCN'08) – 2008, held at SLIET, Sangrur (Punjab).

• Deepti Gupta and Shalini Batra, *"Ontology Life Cycle"* **accepted** at National Conference COIT-2007 held at RIMT-IET, Mandi Gobindgarh (Punjab).

• Deepti Gupta and Shalini Batra, *"Developing Semantic Web Services Using WSMT"* **communicated** to Apeejay Journal of Management and Technology – 2008