Software Engineering Modeling Techniques for Semantic Web based Systems

THESIS

Submitted in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

by

MINAL KAMLAKAR BHISE

Under the Supervision of
Prof. Prabhat Ranjan

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
PILANI (RAJASTHAN) INDIA

2009

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
PILANI (RAJASTHAN)

CERTIFICATE

This is to certify that the thesis entitled **“Software Engineering Modeling Techniques for Semantic Web based Systems”** submitted by Mrs. Minal Kamlakar Bhise, ID. No. 2005PHXF016 for award of Ph.D. Degree of the Institute, embodies original work done by her under my supervision.

DR. PRABHAT RANJAN
Professor, DAIICT
Gandhinagar, Gujarat

Date: April 28, 2009

DEDICATED TO MY GRANDPARENTS

AND PARENTS

MINAL BHISE

ACKNOWLEDGMENTS

I am thankful to Prof. L. K. Maheshwari, Vice-Chencellor, BITS, Pilani for providing me the opportunity to pursue the off-campus PhD of the Institute. I express my gratitude to Prof. Ravi Prakash, Dean, Research and Consultancy Division (RCD), BITS, Pilani for his constant official support, encouragement and making the organization of my research work through the past few years easy.

I thank Dr. Hemanth Jadav, Mr. Dinesh Kumar, Ms. Monica Sharma, Mr. Sharad Shrivastava, Mr. Gunjan Soni, Mr. Amit Kumar and Ms. Sunita Bansal, nucleus members of RCD, BITS, Pilani, without whose cooperation and guidance it would not have been possible for me to pursue such goal oriented research during each of the past few semesters.

I also express my gratitude to the office staff of RCD whose secretarial assistance helped me in submitting the various evaluation documents in time and give pre-submission seminar smoothly.

I thank my Doctoral Advisory Committee (DAC) members, Dr. T.S.B.Sudarshan and Prof. Sunder Balasubramaniam, who spared their valuable time to go through my draft thesis and were audience to my pre-submission seminar in order to provide valuable suggestions that immensely helped in improving the quality of my PhD thesis report.

I would like to thank my supervisor, Prof. Prabhat Ranjan for his constant support and encouragement. His truly scientific intuition has inspired me and enriched me as a researcher. I thank him for all the arguments and discussions I had with him during the thesis work.

I would like to thank Prof. S.C. Sahasrabudhe, Director, DAIICT for facilitating my PhD work at DAIICT. I would like to express my gratitude and respect towards Prof. S.Venkateswaran, Ex Vice Chancellor, BITS, Pilani, Prof. V.S.Rao, Director, BITS, Pilani Hyderabad Campus, and Prof. G.P. Srivastava, Director, IcfaiTech for their constant encouragement and support. I would like to express my sincere gratitude towards Prof. N.Y. Mehendale for generating initial research interest in my mind during my interactions with her at Pune University. I thank Prof. Sanjay Srivastava for all the discussions and arguments I had with him during the course of this work. He has been a constant source of inspiration and support for me. Special thanks to Dr. T.S. Kumbar, Senior Deputy Librarian, Mr. Manoj Nair and other Resource Center staff of DAIICT for making all the resources available to me.

I wish to express my love and respect for my parents; Prof. Vidula and Kamlakar Bhise, who taught me to strike personal and professional balance. I also want to express love and affection for my sisters; Dr. Swati and Radhika, my husband Prof. Sanjay and son Manas for the understanding & endless love, throughout my life. They managed to keep the home schedule intact in spite of my erratic and long working hours. I want to express my affection for my other family members; Neelima, Sripad, Sanjeev, Madhav, and Prachi. I wish to thank my friends, Dr. Poonam and Dr. Radha for keeping me entertained throughout my work at BITS, Pilani and DAIICT, Gandhinagar.

MINAL BHISE

.

ABSTRACT

Traditional information retrieval deals with small, static, homogeneous, centrally located, monolingual document collections. Web information retrieval deals with huge volumes of data which is volatile, heterogeneous, distributed and multilingual. Semantic web offers a great deal of deviation from the way in which the current search engines which are based on the traditional information search theory work and evaluate the retrieved documents. The upcoming semantic web has three basic components; markup languages, ontologies and intelligent agents. Semantic search is ontology based intelligent information retrieval. Detailed investigation regarding the information retrieval nature of traditional and semantic web systems has been carried out. Earlier very less work has been reported in this direction. These two ways of information retrieval differ in two major aspects. Traditional information retrieval is based on detecting occurrence of keywords in the documents while a semantic search looks for matching concepts which may or may not involve the keywords explicitly. Relevancy of the retrieved documents has been judged differently in traditional and semantic web information retrieval systems. While the relevancy of retrieved documents by traditional information retrieval has been measured using recall and precision, there is a trade-off between these two factors and also the actual relevancy depends on the user's perception. Quality of the semantic web information search can be measured using precision, recall and response time. In addition to the usual keywords, indexing keywords and semantic concept level of abstractions, one can visualize relationships among semantic concepts at the highest level of abstraction for the semantic web information retrieval model. Semantic search needs to deduce relationships from the given set of concepts and relationships defined in the domain ontology and further make use of those relationships for inferencing. As the semantic search is based on ontologies for intelligent information retrieval, good semantic search needs a good ontology. The ontology building process is time consuming and difficult and needs domain experts to define basic concepts and structures. The various Ontology Representation Languages like KIF, SHOE, Topic Maps, DAML, OIL and OWL lack good visual modeling tools which are a must for human comprehension of ontologies. Further these visual models

should have an ability to map to machine understandable representations which will be used by the intelligent agents for inferencing and integration. The techniques used so far for knowledge representation are based on Knowledge Interchange Format KIF which has very small following and that too within AI community only. The complexity of the ontology building tools and the lack of good, standard interface for ontology builders make the ontology building very difficult for the domain experts. As a result, the domain experts do not contribute substantially towards ontology building. There is a need for more familiar notations and tools for a uniform representation of ontologies. The OMG's UML being a standard modeling language in software engineering, it is better supported in terms of expertise and the tools as compared to the upcoming semantic web ontology language, OWL. The UML is expressive and standardized modeling language which has large user community and very good commercial tool support in the form of IBM Rational Rose, Magic Draw, JUDE and ArgoUML. Use of UML for ontology representation will allow many mature UML tools, models and expertise to be applied to knowledge representation systems not only for visualizing the complex ontologies but also for managing the ontology development process. UML models are graphical models hence are very easy for human comprehension and for management. The UML based modeling of domain ontologies for semantic web based systems and effect of it on the semantic web information retrieval has been reported in this thesis. The web based systems of the future will consist of smaller, independent systems, each providing access to different contents. All these smaller systems should work together hence interoperability among the systems is the core issue. These smaller systems can work together if they are supported by ontology. One can make such systems available on the semantic web. The semantic web intelligent agents will make such systems more scalable, flexible, extensible, and interoperable. This work offers possible solution in the form of a semantic web based system domain ontology development using UML domain model. The basic functionalities of these agents can be analyzed using use case models. To improve the completeness of the generated ontologies one can make use of the behavioral UML models like interaction and state models. These models can give information about the responsibilities and roles of classes and objects from the messages and interactions. The communication among agents can be visualized in the form of

message passing and analyzed using UML interaction models. State models help to analyze state changes that objects undergo when they participate in processes described by use case model.

Main objective of the presented work is to model semantic web based system using software engineering modeling techniques. This thesis takes up semantic web based digital library as a case study to demonstrate the proposed software engineering modeling technique. Any software system has two aspects to model; structural and behavioural. Various UML models depicting structural and behavioural aspects have been developed for the chosen case study. The domain ontologies and agent behaviour have been modeled for the semantic web based digital library. The effect of semantic web domain modeling on semantic web information retrieval has been discussed. The mappings from software engineering modeling languages to ontology representation languages have been worked out and demonstrated for the semantic web based digital library. The transformation of UML models to Java and OWL code is based on XSLT technology. This mapping technique can help to automate development and maintenance of semantic web based systems.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
BM	Boolean Model
DAML	DARPA Agent Markup Language
DC	Dublin Core
EBM	Extended Boolean Model
IR	Information Retrieval
KIF	Knowledge Interchange Format
LSIM	Latent Semantic Indexing Model
MDA	Model Driven Architecture
MOF	Meta Object Facility
OCL	Object Constraint Language
ODM	Ontology Definition Model
OIL	Ontology Inference Layer
OMG	Object Management Group
ORL	Ontology Representation Language
OUP	Ontology UML Profile
OWL	Web Ontology Language
PIM	Platform Independent Model
PM	Probabilistic Model
PSM	Platform Specific Model
QVT	Query View Transformation
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
SBM	Standard Boolean Model
SHOE	Simple HTML Ontology Extensions
SWIR	Semantic Web Information Retrieval
SWRL	Semantic Web Rule Language
TIR	Traditional Information Retrieval
UML	Unified Modeling Language
URI	Uniform Resource Identifier
VSM	Vector Space Model

XML	eXtensible Markup Language
XSLT	XML Stylesheet Language Transformations

1. Introduction and Literature Survey

Information retrieval (IR) technology is currently being applied to a variety of application domains from database systems to web search engines. The traditional information retrieval (TIR) deals with small, static, homogeneous, centrally located, monolingual document collections. Web information retrieval deals with huge volumes of data which is volatile, heterogeneous, distributed and multilingual. The semantic web offers a great deal of deviation from the way in which the current search engines which are based on the traditional information search theory work and evaluate the retrieval results. The semantic search is ontology based intelligent information retrieval. The detailed investigation regarding the information retrieval nature of traditional systems and semantic web systems has been carried out and reported in this chapter. Earlier very less work has been reported by researchers in this direction. There are two major aspects in which two ways of information retrieval differ; the way in which the information retrieval is carried out and the evaluation of the search answer. The TIR has been based on detecting occurrence of keywords in the documents while a semantic search looks for similar concepts which may or may not involve the keywords explicitly. The relevancy of the retrieved documents has been judged differently in traditional and semantic web information retrieval (SWIR) systems. While the relevancy of retrieved documents by traditional information retrieval has been measured using recall and precision, there is a trade-off between these two factors and also the actual relevance of retrieved documents depends on the user's perception. The quality of the semantic web information search can be measured using precision, recall and response time. In addition to the usual keywords, indexing keywords and semantic concept level of abstractions, one can visualize relationships among semantic concepts at the highest level of abstraction for the semantic web information retrieval model. Semantic search needs to deduce relationships from the given set of concepts and relationships defined in the domain ontology and further make use of those relationships for inferences. As the semantic search is based on ontologies for intelligent information retrieval, good semantic search needs a good ontology. The software engineering modeling technique for visual modeling of ontologies has been proposed in this thesis, which makes the ontology development process more domain expert friendly. This thesis discusses the use of software engineering techniques to model various aspects of the semantic web based system like digital library. The structural and behavioural aspects of the semantic based system have been modeled using Unified Modeling Language (UML) technique. The domain ontology and intelligent agent behaviour of semantic web based digital library have been modeled using UML models. The effect of semantic web domain modeling techniques on SWIR has also been discussed. The mapping from software engineering modeling languages to ontology representation languages have been worked out for the digital library system.

This chapter includes the discussion on TIR, and problems with TIR. It also discusses SWIR in the light of the TIR. The focus of this chapter is to integrate the research efforts made by earlier researchers in the field of semantic search, software engineering modeling technique based domain ontology modeling and research issues in digital library domain.

1.1 Traditional Information Retrieval (TIR)

Traditional Information Retrieval which is keyword based depends largely on pure lexical analysis. Lexical analysis does not have the ability to distinguish between semantically disjoint ideas that happen to share a common lexical expression. The three classical models of information retrieval are Set Theoretic Models, Algebraic Models and Probabilistic Models. The other models like latent semantic indexing based models and generalized vector space models derived from these basic models are in existence [DOM2000]. IR algorithms use various techniques like keyword matching, stemming, latent semantic indexing and relevance feedback to match the keywords represented as bit patterns with the input query bit pattern [BYR1999]. Many traditional IR systems use relevance feedback method to determine the actual relevance of the retrieved documents based on the feedback obtained from users. The relevance feedback modifies the original query to produce a new improved query and as a consequence new ranking of documents. A typical TIR system based on relevance feedback method has been shown in Figure 1. The TIR system tries to retrieve all documents that satisfy the query submitted by the user. These documents are ranked according to their relevance for the query. Queries and documents are usually expressed in natural language and need to be processed by query and document processors. The query and documents are represented as two lists of terms. These terms are often weighed using some weighting schema that is meant to capture the importance of that term in representing that document or query. The document relevancy is determined by the presence or absence of keywords specified by the information seeker.

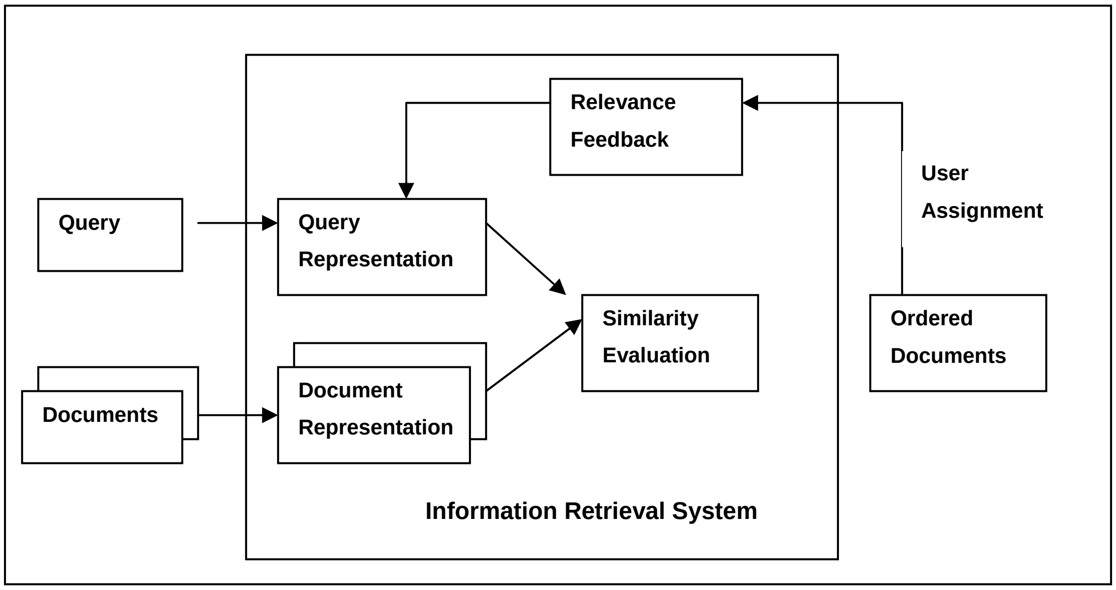


Figure 1 Traditional IR

The statistical approaches break documents and queries into terms which are in fact words appearing in documents. This list of words often undergoes some preprocessing. Stemming of the words is done to extract root of each word [Por1980]. The objective is to eliminate the variation that arises because of different grammatical forms of the same word. For example, approaches, approached, approachable, and approaching should be recognized as forms of the same word ‘approach’. Sometimes this preprocessing uses list of stop words to remove words which are too common to be useful in distinguishing a document.

The Set Theoretic Models like standard boolean model (SBM) and extended boolean model (EBM) represent documents as sets of words or phrases. Similarities are usually derived from set-theoretic operations on those sets. In Boolean model (BM), the documents and queries are represented as sets of index terms each assigned a weight (0 or 1) indicating a presence or absence of that particular keyword. The Boolean query uses classical operators like AND, OR, and NOT. The result of the Boolean query is either true or false ie.document is relevant or irrelevant. Hence it is not possible to rank results of such queries. Also a document containing many terms from query is not treated any better than the document containing no term at all [SB1988]. The EBM takes care of this problem by extending the Classical Boolean model with the functionality of partial matching and term weighting but formulating extended Boolean query is a tough task and hence it is very rarely used. Algebraic Models represent documents and queries usually as vectors, matrices or tuples. The similarity of the query vector and document vector is represented as a scalar value. Some of the models falling in this category are vector space model (VSM), generalized VSM, topic based VSM and latent semantic indexing model (LSIM) [BYR1999]. VSM has been discussed in detail for semantic web based retrieval systems. The term-term correlations are considered in generalized VSM. But it is not clear in which situations the generalized model outperforms the classic VSM. Also the cost of computing the ranking in this model is fairly high as compared to the classic VSM. The ideas in a text are more related to the concepts than the index terms used in its description. The main idea of the LSIM [F+1988] is to map each document and query vector into a lower dimensional space which is associated with concepts. But carrying out information retrieval using this way may not work for general collections of documents. In the Probabilistic Model (PM), the framework for modeling documents and query is based on probability theory. PMs treat the process of document retrieval as a probabilistic inference. Similarities are computed as probabilities that a document is relevant for a given query. Probabilistic theorems like the Bayes' theorem [Mac2003] are often used in these to calculate the probabilities of various terms in these models. Some of the probabilistic models are binary independence retrieval model and probabilistic relevance model. With probabilistic models the ranking of output results may improve but it has been found that guessing initial separation of documents into relevant and non- relevant sets is very difficult task. This method doesnot consider the frequency with which index term occurs inside a document. Also Salton and Buckley [SB1988] showed that the VSM is expected to outperform the PM with general collections. Hence VSM is the most popular model among all IR models.

In VSM, documents and queries are represented as vectors. The components of these vectors are usually terms appearing in documents. Usually stop words are eliminated and the remaining words are stemmed so that only one grammatical form of the root word remains in the final list of vector components. Each component of the vector has been assigned a non-Boolean (anything between 0 and 1) weight. This weight represents an estimate of usefulness of the given term as a descriptor of the given document.

The weighting algorithm used in this model makes use of the term frequency- inverse document frequency *tf-idf* method [BYR1999]. The weight w_{ij} corresponding to the i^{th} component of the document d_j vector representation is given by

$$w_{ij} = tf_{ij} * idf_i$$

Where $tf_{ij} = f_{ij} / \max_l (f_{lj})$. The maximum is computed over all terms mentioned in the document d_j.
idf_i is the inverse document frequency for k_i. *It is given by* $idf_i = \log (N / n_i)$
The relevance ranking is computed as the cosine of the angle between the document vector d_j and the query vector q.

$$Sim(d_j, q) = \cos \Theta = (d_j . q) / |d_j|\ |q|$$

Where Θ *is the angle between* d_j *and* q.

The terms which appear in too many documents (e.g., stopwords, very frequent terms) receive a low weight, while uncommon terms appearing in few documents receive a high weight. This makes sense since too common terms (e.g., "a", "the", "of", etc) are not very useful for distinguishing a relevant document from a non-relevant one.

The partial matching strategy allows retrieval of documents that approximate query conditions. This model is based on the assumption that the index terms or keywords are independent. Even if we consider the term dependency, some of the dependencies may be local and are not true for all the documents in the collection. This particular fact may affect the performance of the model. The search is based on the keywords so it can not address the problem of semantic search.

1.1.1 Problems with Traditional IR Methods

The most popular IR model in TIR is VSM. The order in which the terms appear in the document is lost in the vector space representation. Due to the lack of semantic sensitivity, the documents modeled using VSM with similar context but different term vocabulary are considered different. The problems with traditional information retrieval models have been reported by us recently [Bhi2007a]. The keyword based search of IR method leads to semantic insensitivity of search results. The measurement of relevancy of retrieved results can not be properly done using the precision and recall for web based systems.

1.1.1.1 Keyword Based Search

Traditional IR depends largely on pure lexical analysis to identify interesting or useful information within a large collection. Such analysis focuses on comparing the characters that make up words (in turn words that make up phrases), to determine relevancy of the document. The lack of common ‘keywords’ or terms in two documents do not necessarily mean that the documents are not related. Two terms can be semantically similar even if they are lexicographically different. However, lexical analysis does not have the ability to distinguish between semantically disjoint ideas those have common lexical expression. For example, a search term such as ‘chair’ would return a ‘furniture’ as well as ‘conference’ chair even though the user might be interested in knowing about furniture chairs only. The classical retrieval methods fail to retrieve documents with semantically similar terms. The relevance of the document is determined by the presence or the absence of the keywords specified. The importance or weights assigned to the keywords is based on spatial or proximity based search for some algorithms.

1.1.1.2 Relevance of the Retrieved Document

The relevance of the document is determined by the presence or the absence of the keywords specified. There are two basic measures, precision and recall, used for measuring the relevance of the retrieved document. The precision refers to the proportion of documents retrieved by a query that are relevant to that query. Recall refers to the proportion of relevant documents that exist in the entire document collection that are retrieved by a given query. In case of traditional IR, the increase in precision decreases recall and vice versa. The occasional major mistakes have very little impact on the average recall/precision measures used in standard IR tests, but considerable impact on end users as they may have been looking for a specific document which is very much relevant for the query.

The web users who utilize search engines are not so much interested in the traditional measure of the precision but are interested in knowing how many of those retrieved results appear in the first 8-10 searches on the first page of results. Also there is very little hope of being able to calculate the recall for the web based query, the web user would be concerned about retrieving and able to identify pages which are considered as valuable for the given query. So precision and recall are not very practical measures of document relevance for web search. Usually web users tend to give more importance to performance issues like interactive response time. In web searching there is a trade off among precision, recall, and response time. For web users the recall might be calculated based on the information rich Hub pages for the first 10 or 20 searches [Pok2004]. Ultimately it is only individual user who can determine the relevancy of the retrieved document.

As far as traditional IR systems are concerned, the relevancy of the retrieved document is limited to presence or absence of key terms appearing in the query. The traditional information retrieval can be seen as happening at two abstraction levels namely, actual keywords, indexed keywords. The actual keyword search happens at the document level and looks for the occurrence of actual keywords in the document. The semantic search happens at the highest level of abstraction that is concept level; it looks for matching semantic concepts. In the presented research work, a slight modification to this 3 level semantic search model has been suggested. There is a need to visualize the top most fourth layer consisting of semantic relationships among semantic concepts defined at level 3.

1.2 Metadata Based Information Retrieval

In bibliographic systems, subject headings or descriptors from a controlled vocabulary for the given problem domain have been used by many researchers. By searching on a well designed controlled vocabulary, search results can be improved. However, there can be problems if the terms used in the vocabulary are not known to the user, also problems can arise while incorporating new and changed terminology [Bat1988]. Different types of metadata standards for different domains have been defined over the years like NewsML for news domain, Dublin Core (DC) for digital library domain, MPEG-7 and VoiceXML for media specific information. One can query the database based on the metadata that is used to encode the data. The DC model [You1997] is leading a way in these efforts by providing ways to achieve syntactic information retrieval for digital library domain. DC meta-model based digital library domain specific problems can be handled by a system which is based on certain structural standard. It can handle queries based on DC elements like title, publisher, contributor etc. The metadata based information retrieval system helps in answering the queries beyond usual keyword matching but still the metadata elements is the limit. The present web IR is based on indexing and the SWIR is based on the use of metadata. This system will go beyond keywords in documents and queries, and instead match based on topic, data type, and relations among data and many other qualities which are included in the controlled vocabulary.

The main thing missing from the Web search engines and Commercial IR systems is 'understanding of the query'. The semantic web allows us to pose the queries based on the semantic metadata and ontologies. We can question the document based on the topic/ document type/ author depending upon the metadata standard used to encode the metadata for the document. But this technique does not help much in answering complex queries which possibly need to collect, analyze and integrate the required data from various resources. If one document inserts 'dog food' in the topic meta tag and another document has 'canine nutrition' as a topic then how will an IR system determine whether these two documents are equivalent? This problem can be solved by making the IR system use the ontologies which establish the equivalence of the 'dog food' with 'canine nutrition'. This means the query answering over several documents is a major challenge and so is the automatic creation of metadata.

1.3 Semantic Web Information Retrieval (SWIR)

The semantic web will have to deal with huge, dynamic, heterogeneous and distributed data. The semantic web will enable the user of the web to find the relevant and required information from the web in acceptable time. Semantic web queries have to be mapped to the lowest level of representation in the form of bits that is the way in which this target information is stored in the computer. The queries like "I have to fix up an appointment with a surgeon" have been expected to be answered within reasonable time [BHL2001]. The semantic web architecture, called semantic web stack is shown in Figure 2. The bottom layers are represented as Extensible Markup Language (XML) Namespace, XML Schema, Uniform Resource Identifier URI, and Unicode. The middle layers consisting of Resource Description Framework RDF, RDF Schema, and Web Ontology Language (OWL) represent technologies necessary for building semantic web applications. The top layers consisting of logic, proof and trust have not been standardized yet.

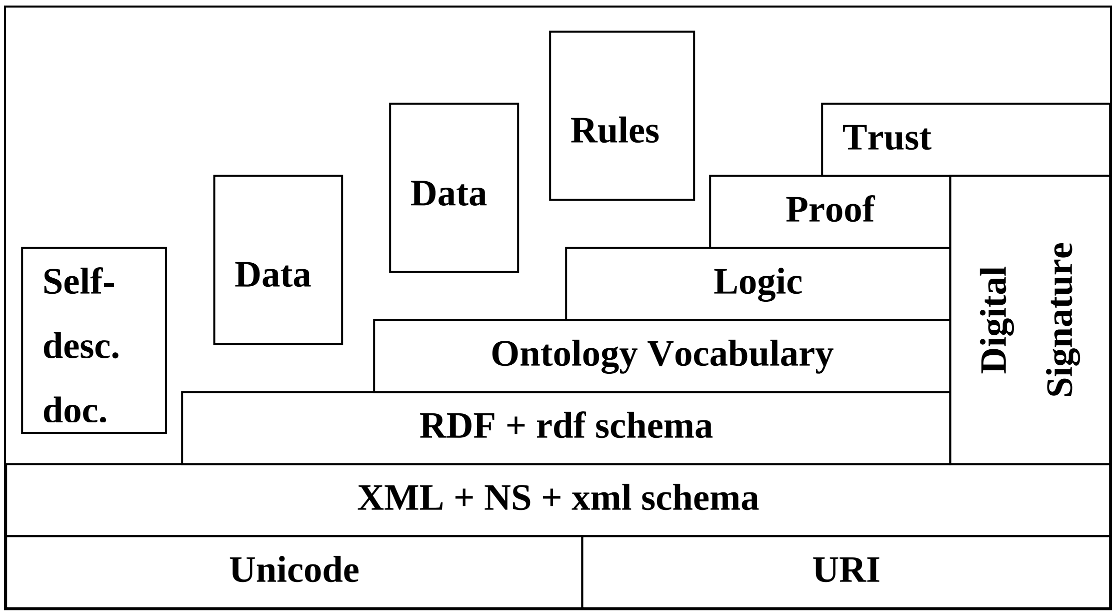


Figure 2 Semantic Web Stack [BHL 2001]

The semantic search is an ontology based intelligent search. Answering complex queries need to collect, analyze and integrate the required data from various sources. Semantic web search addresses this issue by developing a semantically aware system which makes use of ontologies to integrate the information from various sources. The most popular semantic web search techniques like vector space model based similarity match and hybrid spread activation have been discussed in detail in this thesis. The presented work integrates the research efforts made by earlier researchers in the field of semantic search.

1.3.1 Role of Semantic Web Components in Information Retrieval

The semantic web has three components; Ontologies, Markup Language, and Intelligent Agents. Each of the semantic web components is related to the domain knowledge representation in some way. Together they form an intelligent information retrieval system which is ontology based.

1.3.1.1 Ontology

Ontology is a set of logical axioms designed to account for the intended meaning of a vocabulary. Ontology is an explicit specification of a conceptualization [Gru1993]. Ontology specifies the meanings of the terms in a given vocabulary by explicitly defining the relationships between the terms. By defining shared and common domain information, ontologies can help people and machines communicate concisely supporting semantics exchange, not just syntax [BHL 2001]. The ontology building process needs standard interfaces and tools for creating and updating these ontologies. The software engineering based modeling technique discussed in this work addresses this problem. Guarino [Gua1998] names several research fields in computer science that have embraced ontologies, including knowledge engineering, knowledge representation, language engineering, database design, information retrieval and extraction. Semantic web relies on ontologies for global scale knowledge management.

1.3.1.2 Markup Languages

A markup language is a tool for adding information to the documents. Semantic markup is expected to have universal expressive power as well as syntactic and semantic interoperability. Syntactic interoperability means how easy it is to read the data and get a representation that can be exploited by applications. It is about parsing the data. Semantic interoperability means defining mapping between unknown terms and known terms in the data. The XML does not provide semantic interoperability. The XML allows everyone to create their own tags. The scripts or programs can make use of these tags. The script writer has to know what the page creator uses each tag for. Using XML as a base, a number of new markup languages have been developed to meet semantic interoperability, these are Resource Description Framework (RDF), RDF Schemas (RDFS), Simple HTML Ontology Extensions (SHOE), DARPA Agent Markup language Ontology Interface Layer (DAML+OIL) and recently OWL [Dac2003]. The work done earlier in domain knowledge representation using software engineering modeling techniques was concentrated towards developing the mapping of the software engineering models to the RDF, RDFS and other markup languages [Cra2001a]. At present the researchers are trying to map the domain models to the OWL representation [DGD2004].

1.3.1.3 Intelligent Agents

Ontologies and markup languages will make the semantics of the documents available to the machines. Software agents will make use of the semantic content which is in the form of ontologies, logic, trust, and proof to actually interpret and integrate the content of documents (and queries) to perform tasks for users. Agents will have to carry out the task of finding out and using the information on the web from several resources on their own, processing them and integrating the results and presenting them to the users or carrying out inferencing based on those results. So the different ontologies should be able to work together in order to make all this possible [Hen1999]. Therefore we need a standard way of representing these ontologies which are understandable by human as well as semantic web agents for automatic processing.

Answering complex queries need to collect, analyze and integrate the required data from various sources. Semantic search addresses this issue by developing a semantically aware system which makes use of ontologies to integrate the information from various sources. Therefore we need a standard way of representing these ontologies which are understandable by human as well as agents for automatic processing [Cra2001a, Bhi2007b].

1.3.2 Ontology Based Information Retrieval

The various semantic web search techniques like VSM based similarity match and hybrid spread activation have been discussed in detail by researchers [PK2003, RSA2004]. The focus of this section is to integrate the research efforts made by earlier researchers in the field of semantic search.

The idea behind the semantic web is to augment web pages with mark-up that captures some of the meaning of the content on those pages. Automatic tools can collect and understand the knowledge annotated on a page, and ontologies help make such mark-up compatible across various information sources and queries. So semantic web can answer complex queries which need to collect, analyze and integrate the required data from various resources. Language is the framework for transferring meaning between people.
The semantic web depends on a framework for transferring meaning between computer programs. Several mark-up languages have been used to communicate between programs on the semantic web like XML, RDF, RDFS, and OWL. A markup language is a tool for adding information to the documents.

- XML : language defines the syntax for other semantic web languages.
- RDF (Resource Description Framework): Provides the grammar, the basic sentence structure (subject, predicate, object)
- RDF Schema: Allows you to define the nouns and verbs (i.e. Classes and Properties)
- OWL (Web Ontology Language): OWL Builds on RDF and RDFS. It is more expressive. This has been accepted as a standard markup language for semantic web [DS2004].

All these play a definite role in making semantic search possible which is an intelligent search based on ontologies. The XML can support only syntactic search while OWL makes possible semantic search that we are interested in. Semantic markup OWL has universal expressive power, syntactic and semantic interoperability. Intelligent agents will have to carry out the task of finding out and using the information on the web from several resources on their own, processing, integrating and presenting the results to the users or carrying out inferencing based on those results. So the different ontologies should be able to work together in order to make all this possible. By defining shared and common domain information, ontologies can help people and machines communicate concisely supporting semantics exchange, not just syntax [BHL2001].

The semantic web information retrieval system is based on a domain knowledge representation schema in the form of ontology. Use of ontology enables to define concepts and relations representing knowledge about a particular document in domain specific terms. New resources registered within the system are linked to concepts from this ontology. The resources may be retrieved based on the associations and not only based on partial or exact term matching as the conventional vector model presumes.

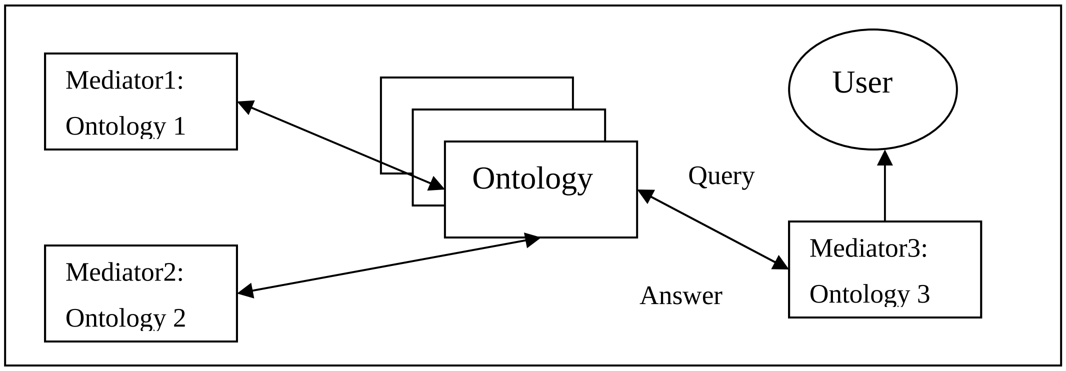


Figure 3 Ontology Based Information Retrieval [Bhi2007a]

Figure 3 depicts the architecture for ontology based information retrieval. A user formulates a query in *Ontology 3*; a mediator then transforms this query to set of queries based on other ontologies and their distribution.

One of the earliest works on SWIR [SFJ2002] suggests that addition of semantic markup to web documents will make possible to perform inference over document content. The traditional searches are done with the help of the indexing based on the keywords. The techniques used for indexing are stemmed words, characters n-grams, and overlapping sequences of n contiguous character in this work. It has also suggested the use of semantics for indexing terms for a traditional search engine. For example, each distinct OWL tag can work as indexing term. This method would try to exploit the statistical associations between semantic markup and text. Rocha [RSA2004] presents a search architecture which combines classical search techniques with the spread activation techniques applied to a semantic model of a given domain. The strength of the relationships was determined based on certain ontology properties. The spread activation techniques was used to find related concepts in the ontology given an initial set of concepts and corresponding initial activation values from the classical search techniques applied to the data associated with the concepts in the ontology. The system infers relations through a spread activation algorithm, making it possible to retrieve concepts which do not contain any of the specified words. It has been found that the most popular SWIR algorithms are a combination of vector space model with the spread activation technique.

1.3.3 Use of Vector Space Model for Ontology Based Semantic Information Retrieval

In semantic search, the role of keywords is played by concepts from domain ontology. Hence the documents at semantic level could be represented as vectors in a hyperspace defined by the set of all ontology concepts. The weight assigned to a concept represents the relative importance of that concept in the problem domain. Queries will be represented as weighted concepts where the weight of each component defines user's level of interest in a particular concept modeled by the ontology.

The appropriate query interfaces are required in order to capture the user needs and convert them to the corresponding vector representation. The ontology concepts are related by different kinds of relationships like inheritance, aggregation and are not orthogonal as presumed by the vector model.

1.3.4 Hybrid Spread Activation Approach to Searching in Semantic Web

One possible approach for semantic search is to combine the usual traditional search with the ontology based information retrieval [RSA2004]. It consists of a combination of classical search technique with the spread activation technique. For the semantic web domain ontology, the weights are assigned to links based on certain properties of the ontology representing the strength of the relation. The spread activation technique is used to find related concepts in the ontology given some initial set of concepts and initial weights. These initial weights have been found out using classical search [GMM2003].

A semantic search technique which enables user to express the information needs in terms of keywords has been discussed in this work. Further search is based on the semantic information of the domain to obtain results that are not possible with the traditional search.

1.3.4.1 Weighting Algorithm

In traditional IR *tf-idf* strategy [BYR1999] the first measure gives the degree of similarity between two related concept instances in a relation and the second measure gives the specificity of the concept relation. This Cluster measure for concept instances C_j and C_k is given by:

$W\,(\,C_j,\,C_k) = \sum \{\sum n_{ijk} / \sum n_{ij}\ \}$

Where *nij* represents that concepts C_j and C_i are related and n_{ijk} represents that both the concepts C_j and C_k are related to concept *Ci.* Therefore $(C_j,\,C_k)$ represents percentage of concepts that C_k is related to that C_j is also related. This particular measure reflects the fact that concepts sharing common relations are semantically similar.

The Specificity measure is given by:

$W\,(C_j,\,C_k)\ =\ 1/\sqrt{n_k}$

Where n_k is the number of instances of given relation type that have *k* as its destination node. The actual measure is the product of cluster and specificity measures.

1.3.4.2 Spread Activation Algorithm

The relations or links have both, a label coming from an ontology definition and a numerical weight assigned using weight mapping technique described in previous section. Given an initial set of concepts, the algorithm obtains a set of closely related (semantically related) concepts by navigating through the linked concepts in the graph. This algorithm is domain dependent. The Constrained Spread Activation applies constraints like maximum path length, fan-out etc to propagation [Cre1997].

The algorithm has as a starting point, an initial set of instances in the ontology with each having an initial activation value. The initial nodes will be put in priority queue, in decreasing order of their activation values. The first one from the queue is processed. If it satisfies certain conditions, it propagates its activation to its neighbors. The neighboring nodes which are not there in the present queue will be added to the queue. The processed node is added to the list of processed nodes.

The ontology concepts depicted as vectors in vector model are related by different kinds of relationships like inheritance, aggregation, association; and are not orthogonal as presumed by the model. There are two important aspects of semantic search namely, good domain ontology and understanding the semantic relationships between ontology concepts [BYR1999, PK2003]. The inheritance relationship relates concepts with more specialized sub-concepts. Any other complex relationship among concepts can be represented by introducing appropriate operators. The semantic information retrieval has been seen so far by researchers as happening at three abstraction levels namely, actual data at level L1, keywords at level L2 or indexed keywords and semantic concepts at level L3 as shown in Figure 4. The semantic search happens at the highest level of abstraction that is the concept level; it looks for matching semantic concepts. In the presented research work, a slight modification to this 3 level semantic search model has been suggested as depicted in Figure 4.

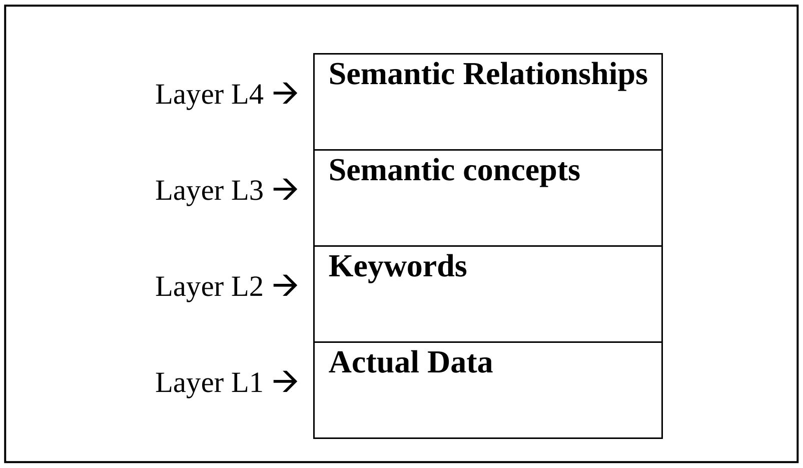


Figure 4 Modified Semantic Web Information Retrieval Model [Bhi2009a]

There is a need to visualize the top most fourth level L4 consisting of semantic relationships among semantic concepts defined at level 3. The semantic search needs to deduce relationships from the given set of relationships defined in the domain ontology and further make use of those relationships for inferences.

1.4 Ontology Modeling Techniques

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 [BHL2001]. Semantic web agents will have to carry out the task of finding out and using the information on the web from several resources on their own, processing them and integrating the results and presenting them to the users or carrying out inferencing based on those results. So the different ontologies should be able to work together in order to make all this possible [Hen1999]. Therefore we need a standard way of representing these ontologies which are understandable by human as well as semantic web agents for automatic processing. The present ontology building tools do not have standard visual interfaces. The ontology building tools are mainly built on AI principles hence are very difficult to use outside AI community. The domain experts are expected to play a major role while building ontologies. They usually define the terms and structures which are integral part of the problem domain. The AI based ontology tools also restrict the contribution of domain experts in the ontology building process.

Building ontology consists of acquiring domain knowledge; assembling appropriate information with consensus and consistency in the form of terms used formally to describe things in the domain of interest. The domain experts then organize ontology which involves forming conceptual structure of the domain concepts and their properties, identifying relationships among concepts, creating abstract concepts as organizing features, referencing or including supporting ontologies. In general the ontology development process demands lots of involvement from domain experts. The ontology building process is not a linear process but an iterative process. Skeleton structure of core concepts is extended with more refined and more peripheral concepts.

The procedural and object- oriented software uses structural aspects of the software to control program flow and use. Ontology languages primarily use structures to specify semantics. For example, the subclass inheritance in object-oriented languages is a mechanism of convenience that facilitates code organization and reuse, the same in ontology language enables semantic interpretation of the data through classification and restrictions. An ontology building process spans problem specification, domain knowledge gathering and analysis, conceptual design and commitment to problem domain ontologies. All these processes are iterative in nature. The nature of ontology development process fits well in the well known object oriented way of developing systems. This thesis addresses the semantic web ontology development issue by using Meta- Object Facility MOF [B+2001a] based object oriented paradigm of developing systems.

Some ontology development tools can automate portions of this ontology building process. The survey of the existing ontology editing tools [Den2005] suggests that higher level of abstraction of ontology constructs is desirable as it will allow more intuitive and powerful knowledge modeling process. The present ontology tools also lack easy navigation. The reasoning abilities to explore, compose, and check ontologies and also the facilities to integrate different ontologies are needed. The need for standardization of ontology modeling tools also has been a concern. Ontologies are for sharing so they help to exchange and interpret information. The wider range of applications and other ontologies must be able to use the created ontology for its greater utility and utility of the interrelating ontologies. The traditional approaches to ontology representation use modeling formalisms developed by the artificial intelligence knowledge representation community, such as Knowledge Interchange Format KIF [Fik1994] ontology representation language and Description Logics [B+2002]. These languages were developed for use in monolithic knowledge representation systems that are very different in character from distributed multi-agent systems like semantic web [Hor2002]. KIF provides a Lisp-like syntax for expressing sentences of first order predicate logic and also provides extensions for representing definitions and meta knowledge. The Stanford University Knowledge Laboratory developed Ontolingua

[F+1995] which is based on KIF. Description Logics are a family of logic based Knowledge Representation formalisms descended from semantic network, frame-based systems and KL-ONE [BS1985]. They are designed to focus on concepts and roles. In Description Logics, concepts are sets of similar objects and roles are binary relations between objects. There are several classical description logics based systems, such as LOOM [Loo], GRAIL [R+1997], CLASSIC [BB1989], and KL-ONE. The various Ontology Representation Languages (ORLs) like KIF, SHOE, Topic Maps, DAML, OIL and OWL lack good visual modeling tools which are a must for human comprehension of ontologies [CP1999]. Further these visual models should have an ability to map to machine understandable representations which will be used by the agents for inferencing and integration. The techniques used so far for knowledge representation are based on KIF which has following within AI community only [Fik1994]. The complexity of the ontology building tools and the lack of good, standard interfaces for ontology builders make the ontology building very difficult for the domain experts [SHB2006]. As a result, the domain experts do not contribute substantially towards ontology building. UML being a standard modeling language in software engineering, it is better supported in terms of expertise and the tools as compared to the upcoming semantic web ontology language, OWL. The UML is expressive and standardized modeling language which has large user community and very good commercial tool support in the form of IBM Rational Rose, Magic Draw, JUDE and ArgoUML [Den2005]. The use of UML for ontology representation will allow many mature UML tools, models and expertise to be applied to knowledge representation systems not only for visualizing the complex ontologies but also for managing the ontology development process. The UML models are graphical models hence are very easy for human comprehension and management.
There is use of formal software engineering modeling languages like Z, Alloy reported for checking and reasoning the semantic web ontologies [DSW2002, DSW2003]. The ontology modeling using Z makes the inferencing feature of semantic web powerful. The intelligent web that we are talking about borrows many elements from the artificial intelligence area. The work regarding role of software engineering techniques for analysis, design and development of intelligent systems has been reported recently [DGD2004, DGD2007].

1.4.1 Software Engineering Techniques for Ontology Modeling

The first major attempt of using UML for Ontology Representation is by Cranefield [CP1999, Cra2001a]. His work points out various reasons for using UML for ontology representation in his work. The UML class diagram offers the static modeling capability for ontology representation; on the other hand the object diagram is suitable for modeling the instances. Object Constraint Language, OCL, is used for the constraint representation in UML. Cranefield created an XMI file from the UML representation. The XSLT style sheets for Java and RDF then produced Java classes along with interfaces and the RDF representation for the corresponding ontology. He uses RDF bags and sequences to transform the unordered and ordered association ends. One of the main problems was while the properties in RDF are first class modeling elements; the corresponding associations in the UML cannot exist without classes. Expressing restrictions was also a problem for representation of complex ontologies using standard UML. Our work uses OCL to express constraints. Baclawski [B+2001a, B+2001b] reports that mapping of the UML associations to the corresponding ORL properties is the most challenging task. The UML meta-model makes UML more suitable for representing DAML ontologies. This work identifies the similarities and differences between UML and DAML with illustrations. DAML+OIL property is a first class modeling element while the corresponding UML association is not. To reconcile this difference, a modest extension to the UML infrastructure is proposed. The metamodel MOF (Meta-Object Facility) is introduced. Transformational approaches [Fal2003, NCL2006] are a promising way of establishing a connection between UML and web-based ontology languages. Different proposals to handle the conceptual differences between these languages are compared in Falkovych's work [Fal2003]. The property mapping is based on the extended UML suggested by Baclawski [B+2001b]. Every property that corresponds to a certain kind of an association in UML diagram is defined as a sub-property of a corresponding association type. Association ends are distinguished using association type with name or role name. Djuric [DGD2004, DGD2005] reports that it is difficult to express the description logic concepts borrowed by semantic web (RDF, OWL) using UML. The use of UML Metamodel, and the Ontology Definition Model (ODM) is proposed. The Ontology UML Profile (OUP) model can be mapped to the OWL profile using UML Metamodel ODM. The OUP enables graphical editing of ontologies using UML diagrams. It is based on the basic UML constructs that are customized and extended with new semantics using stereotypes, tag definitions, tagged values, and constraints. These OUP ontologies are converted to corresponding OWL representations using XSLT. ODM should provide for specification of metadata to describe context and scope of the development and the intended use of that ontology. Initial ODM submissions to OMG are by IBM, DSTC, and Sandpiper Software Inc. The UBOT and DUET projects [GB2004. K+2002] use UML front-end for visualizing and editing DAML ontologies. The prototype UML profile for DAML is used. The DUET tool is based on Rational Rose add-ins and a UML profile for DAML.

1.4.2 Research Issues

Apart from the general strategy for transforming the UML diagrams, the following are some of the open research issues regarding modeling techniques that we have tried to tackle in the presented work.

1.4.2.1 Implementation

How to implement these transformations? Answer to this problem may be, taking an advantage of XML encoding of both XMI and DAML one can write an XSLT file that maps the two models [Cra2001a, Cra2001b] But XSLT is very cumbersome when used for complex ontologies. The projects CODIP [CD2001] and UBOT have developed modules for UML to DAML mapping. But these tools are very new and may not be robust. We have used MOF based UML to map UML domain models to the corresponding java and OWL mapping for the digital library domain using XSLT for OWL [DGD2004].

1.4.2.2 Reasoning

Rational Rose offers basic services for checking the UML models for syntactic errors. The range of support for correctness checking of a model is substantially wider for a formal language. We have used MOF based MDA and UML modeling languages to model digital library domain in this work which has been known to facilitate the inferencing.

1.4.2.3 Rules

UML uses OCL to express constraints. Complex ontologies need OCL. Some work is going on to introduce formal semantics for OCL [BRJ1999]. There is proposal for developing rule language for ORL. For Semantic Web it is called SWRL (Semantic Web Rule Language) which is a combination of OWL and RuleML [GB 2004]. Our work does not address this issue as it is not based on any of the formal languages.

1.4.2.4 Reversible Transformation from UML to ORL

The existing transformations from UML to OWL and vice versa are not reversible. The reversible transformations will allow the modeling and inferencing in parallel for efficient ontology development. The modeling tool has to support transformations from both class diagrams and OCL as OCL is needed to represent complex real life ontologies. We have tried to automate the forward transformation of domain models to OWL and Java representations. The XSLT mapping of UML models makes this possible. An ontology modeling tool, Protégé [Pro]**,** also generates the forward transformation of input ontology to owl representation. But Protégé lacks good visual interface and hence is not very useful for domain experts developing domain ontologies. Although Protégé claims to work with multiple ontologies, creating such ontologies is not an easy job using Protégé.

1.4.2.5 Use of UML Metamodel for Ontology Representation & Management:

The Object Management Group’s MDA is based on UML and related standards such as MOF and XMI. The MDA is driving UML to become more and more formal and machine-processable so that models can be used at compile time and runtime and not just as a graphical notation for human to human communication [DGD2005]. Therefore the complex ontologies can be represented using MDA. Also these ontologies will have better inferencing capabilities. The UML and MOF with formal semantics will make UML more suitable for ontology representation. The complex and interoperable ontologies in turn can allow agents to collect web content from diverse sources, process the information and exchange the results with other agents. Not much work has been done in this area so far. The information retrieval aspect of meta-modeling has not yet been explored. Our work uses MOF based MDA and UML to develop domain ontology for the digital library domain. Domain ontologies are going to play very important role in the semantic web applications as SWIR is ontology based.

OWL and MDA [KWB2003] technologies are being developed in parallel, but by different communities. MDA separates implementation details from business functions. Thus it is not necessary to repeat the process of modeling an application or system functionality and behaviour each time a new technology comes along. A complete MDA specification consists of a platform independent model like UML plus one or more PSMs (Platform-Specific Models). The problem of transformation between ontology and MDA-based languages is solved using XSLT.

1.4.3 Software Engineering Modeling Techniques for Agent Behaviour Modeling

Scalability is an important aspect of any semantic web based system. The intelligent semantic web software agents will offer a way to achieve this scalability. The basic functionalities of these agents can be analyzed using use case diagrams. The use of use case model to represent agents and their functionalities has been reported earlier [CCF2000] in the context of agent based software design. The functionalities from user’s perspective are represented using Use Case Model. This work represents functionalities in the form of use cases, actors invoking them and their relationships for foraging robots [AMc1987]. The agent behaviour can be represented and analyzed using behavioural UML models like sequence and state models.   These diagrams can give information about the responsibilities and roles of classes and objects from the messages and interactions. Interaction Model describes the responsibilities of agents, the services it provides, associated interactions, and relationships between agents. The model helps in representing and analyzing communication between agents and other system components. The work on agent-oriented software design [Bau2003, OPB2000] demonstrates use of sequence interaction models for modeling agent communication protocols for BDI agents. There is use of sequence diagrams to model web services reported [Bau2004]. The UML based representation for agent interaction protocols exists [OPB2000]. Most of the work done earlier is for agent based software systems but not much work has been reported for modeling semantic web based systems as far as use of behavioural UML diagrams is concerned. Most of the work done for semantic web so far is centered on use of class diagram for ontology representation. But UML is much richer beyond class diagram. Can other diagrams be used for looking at other aspects of the ontologies? The different knowledge aspects of the agent based systems can be looked at using the other UML diagrams as well. A proposal exists for modeling DAML-S using other UML diagrams [EKS2000]. This thesis uses behavioural UML models like interaction and state models to analyze the intelligent agent behaviour for the semantic web based digital library. The details of the behavioural UML models are there in Chapter 2.

1.5 Digital Library Research Issues

With the advent of the internet and semantic web, more and more web based applications have been developed. There are health care domain applications [IVB2006], digital library applications [S+1999, K+2000], and many more such applications which are being developed at present. For all such systems information retrieval problems have become more urgent. There are various research issues that exist for semantic web based systems. Scalability is one of issues because we want large pool of information in the form of collections of documents over semantic web usable at human scale. Interoperability problems exist due to significant variations in database formats and structures, richness of information media, multilingual content because of the distributed nature of semantic web based systems. The system must deal with adaptability issue because of the dynamic content and dynamic end users for web based systems. The data on the semantic web will be heterogeneous data having different types of data representations. The semantic web based systems will handle interoperability issues by use of ontologies and scalability issues can be sorted out using intelligent agents. To demonstrate the software engineering based technique, UML, proposed in this thesis, a case study of semantic web based digital library has been worked out. The digital library research issues subsection specifically discusses the research issues for semantic web based digital libraries.

The digital library has emerged as an important application area on the internet. Hundreds of digital libraries are in place around the world and hundreds of digital library projects have been going on. Different user communities need different digital libraries to satisfy their needs. Technologies must be developed to search across these libraries transparently, handling any variations in protocols and formats. This is called syntactic interoperability. Distinct from traditional libraries, digital libraries process large collections of digital objects and provide on-line information services. Typical usage scenarios of semantic technologies in digital libraries include among others user interfaces and human-computer interaction (displaying information, allowing for visualization and navigation of large information collections), and user profiling. Although in the future there would be large number of digital repositories, a digital library system should provide a consistent view of these digital libraries. From a user's perspective, they should appear to be a single digital library system. Many existing digital libraries lack interoperability connections to other libraries. The ultimate goal of digital library research is to achieve semantic interoperability – the ability of a user to access, consistently and coherently, similar (although autonomously defined and managed) classes of digital objects and services, distributed across heterogeneous repositories [Erw1996]. Attention to semantic interoperability has prompted several projects in the Defense Advanced Research Projects Agency (DARPA) funded large scale DL Initiative to explore various artificial intelligence, statistical analysis and pattern recognition techniques. Examples include concept spaces and category maps in the Illinois project [S+1999] and word sense disambiguation in the Berkeley project [Wil1996], voice recognition at Carnegie Mellon project [W+1996], and image segmentation and clustering in the project at the university of California at Santa Barbara [MM1996]. In the NSF workshop on Distributed Knowledge Work Environments; DLib in 1997, a panel of DLib researchers suggested four broad research directions for the digital library; scalability, interoperability, adaptability and durability, and support for collaboration. The result from the semantic analysis could be represented in the form of semantic networks, decision rules and predicate logic. Many researchers have attempted to integrate such results using human created knowledge structures like ontologies, headings and thesauri [McH1990]. Spreading activation based inferencing methods often are used to traverse various large scale knowledge structures [CN1995]. Some of the well-known digital library related research areas are classification, interoperability among heterogeneous collections, communication protocols and standards, search engines, information visualization, usability, and user interfaces related issues [F+1995].

With a lot of attention paid to the study of how to make a better digital library, very little focus has been on simplifying the process of building a digital library. This thesis demonstrates modeling of semantic web based digital library using UML. Realizing the critical role of digital libraries, the projects to build DLib started in 1994 [LOC1995, HMS1996]. The Open Access Initiative (OAI) works on the DLib systems which are easily extendible [SF2002]. Digital libraries are huge and complex information systems which need models and theories. With formal models and theories the researchers are able to describe, specify, and understand complex systems precisely but when it comes to actually building systems using formal paradigms the domain experts can contribute very little. The range of the content and services that are possible for a digital library are potentially very large hence there is no single, complete digital library solution. If we want to make digital library available on the semantic web, we need to develop ontologies which will make the classification, indexing, and interoperability among heterogeneous resources possible. The ontologies and semantic agents will make such digital libraries more scalable, flexible, extensible and interoperable. These ontologies will help intelligent agents to access, retrieve, and integrate the information from scattered sources, analyze them and make decisions. At JCDL (Joint Conference on Digital Library) 2002**,** the research directions for the digital library domain for the next

10 years were discussed. The digital library technical research will make possible high quality, semantically rich, comprehensive information collections, usable for long periods of time. To achieve this, the technology for ontology building was identified as a major hurdle. The ontology building tools available lack good visual modeling interfaces. There are several attempts reported for modeling the digital library domain in a formal way. All these models lack substantial participation from the domain experts hence the quality of the ontologies built are not good. The interoperability and classification issues were not addressed clearly in the research [Wan1999, K+2000]. There are several XML modeling tools like Stylus Studio XML editor, Oxygen available which are only text editors with tree views. The XML modeling tools provide visual interfaces for users. These tools load DTD or XML schema and use the structure information to help a user build an XML instance of that DTD or XML schema. The actual text is not available to the users for editing [SpL2002]. The XML modeling tools have semantic limitations because they produce syntactic models and hence make possible the intelligent agents to work only at syntactic level. The high demand for building digital libraries requires a simplified modeling process and rapid generation of digital libraries. A visual modeling tool would be helpful to domain experts so they can contribute substantially to digital library development. The work reported by Goncalves has proposed 5S theory [G+2002]. 5S represents Streams, Structures, Spaces, Scenarios, and Societies. The Streams Model specifies the communication content between digital libraries and users. The Structures Model specifies how to organize information in usable ways. The Spaces Model specifies how to present information in retrievable and usable ways. The Scenarios Model specifies available information services. Finally, the Societies Model specifies how the digital library satisfies users' demands for information. This approach was a bit too complicated and was not very easy for the domain experts to follow for building digital library [Z+2003].

The work presented in this thesis offers a possible solution in the form of a digital library domain ontology development using MOF based UML domain model. The semantic web software agents will offer a way to achieve the scalability. They make possible construction of distributed, intelligent, robust and scalable system. The basic functionalities of these agents can be analyzed using use case diagrams. To improve the completeness of the generated ontologies one can make use of the behavioral UML models like interaction and activity diagrams. These diagrams can give information about the responsibilities and roles of classes and objects from the messages and interactions. The communication among agents can be visualized in the form of message passing and analyzed using UML interaction models. Very less work has been done by earlier researchers in this direction. We have used UML models like interaction and state models to model semantic web agent behaviour for digital libraries.

The software engineering modeling technique used for modeling semantic web based systems in this thesis is presented in Chapter 2. It includes details of structural and various behavioral UML models. The chapter also discusses suitability of UML models for semantic web based systems modeling which includes mappings from UML models to OWL ontology language. Chapter 3 presents the research methodology used to model semantic web based systems. It includes discussions about XML stylesheet (XSLT) used for transformation of UML models to the Java and OWL codes. Chapter 4 documents use of software engineering modeling techniques for domain modeling of semantic web based digital library. Chapter 5 includes development of various ontology and agent behaviour models. Chapter 6 contains discussions about how automation of semantic web based systems has been achieved using software engineering and XLST techniques. Discussion and Future Work are presented in chapter 7 which contains limitations of the approach presented and demonstrated in the thesis. Chapter 8 is conclusion.

2. Software Engineering Modeling Techniques and Domain Ontology Modeling

This chapter discusses software engineering modeling technique, UML, which is proposed to be used to model semantic web based systems. Various structural and behavioural UML models are discussed in this section 1. Suitability of UML for modeling semantic web ontologies has been critically discussed in section 2. It includes discussions about comparable UML and OWL modeling constructs with their corresponding mappings. Process of ontology and UML class building is discussed in section 3. Section 4 contains discussions about suitability of UML behavioural models like interaction and state for modeling intelligent agent behaviour for semantic web based digital library.

2.1 Software Engineering Modeling Techniques

There has been an increase in the abstraction level of the models. Modeling has become more and more separated from underlying platform hence making their development easier by domain experts. The OMG's Model Driven Architecture is one such effort which defines MOF which further allows defining models formally. The Model Driven Architecture MDA offers a formal structure in the form of meta metamodel MOF hence will be able to express the constraints better. MDA defines three levels of abstraction; Computation Independent Model CIM, Platform Independent Model PIM and Platform Specific Model PSM as shown in Figure 5. MDA is based on 4 layer metamodeling architecture and several OMG standards. These standards are Meta-Object Facility MOF [BRJ1999], Unified Modeling Language UML [BRJ1999] and XML Metadata Interchange XMI [XMI02]. The layer M3, MOF, defines an abstract language and framework for specifying, constructing and managing technology neutral models. The main aim behind having four layers with common MOF layer is to support multiple metamodels and models, to enable their extensibility, integration and conversion possible. We can define UML using the Meta Object Facility MOF available in MDA at M2 layer, a graphical modeling language for specifying, visualizing, and documenting software designs. The meta-meta model Meta-Object Facility MOF [Fre2003] follows four layer architecture, which is applied to define UML and domain specific languages is shown in Figure 6. The top most level (M3) defines the MOF itself. The language specification like UML is at level M2. The model level M1 contains concrete models like class diagram, state diagrams, the notation for which is defined by metamodels at M2. The M0 defines real world objects. It is possible to extend the UML semantics with UML Profiles to suit the requirements of the problem domain [DGD2007] at M2 layer. The technologies for dealing with object networks and graphs cover querying and transformational approaches. Such languages for transformation, querying, and viewing within MOF framework have been proposed within OMG and are called as Query-View-Transformation QVT proposal.

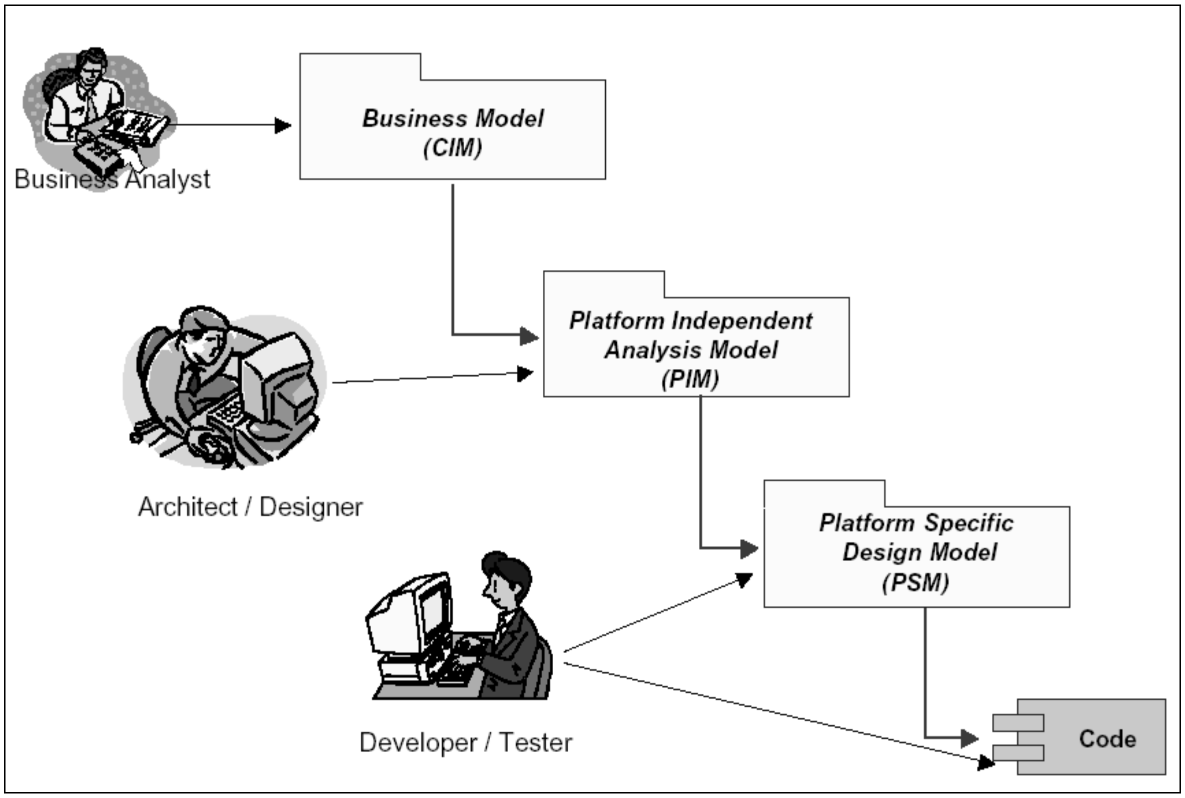


Figure 5 Model Driven Architecture [KWB2003]

The MDA is driving UML to become more formal and machine- processable so that models can be used at compile time and run time and not just as a graphical notation for human-to-human communication. The XMI standard of OMG can serialize the MOF based metamodels like UML Profile and models like UML into plain XML text which is readable by any platform specific implementation. This makes exchange of metamodels and models possible. Thus it is not necessary to repeat the process of modeling an application or system functionality and behaviour each time a new technology comes along. Also it is driving UML more and more formal, so it helps in expressing complex constraints and automatic inferencing by agents. But still no commercial MDA tools are available which can process the models at M3 and M2 layers. The existing UML tools support the MDA models till M1 layer very well [DGD2005]**.** The work reported in this thesis thus models the semantic web based digital library using M1 layer models.

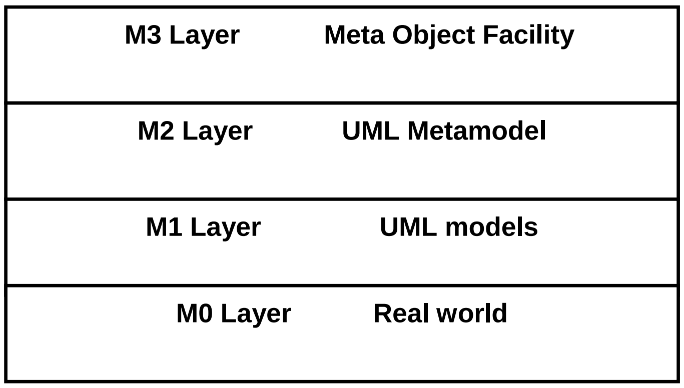


Figure 6 MDA Based MOF Meta Meta-model

2.1.1 Unified Modeling Language UML

UML is a graphical modeling language. Graphical models are more expressive as compared to usual textual models. UML is a set of models or diagrams that can be used to model the static and dynamic aspects of a system. We can model real life complex systems using UML. The structural aspect of a system can be modeled using class model which depicts the classes and their relationships. To model behavioural aspect of a system, several behavioural UML models are available. The classes work in collaboration with each other to achieve certain tasks described in the use case model of the system. The related classes collaborate with each other through message passing. The message passing can be modeled using interaction models namely, sequence and collaboration models. These two models are interconvertible. The objects undergo state changes in response to set of events they are experiencing. The state changes of objects can be modeled using state transition model. The overall flow of activities is depicted in activity model. The package diagram is other structural diagram. Graphically all the models are collections of arcs and vertices. The responsibilities of various UML models have been shown in Figure 7.

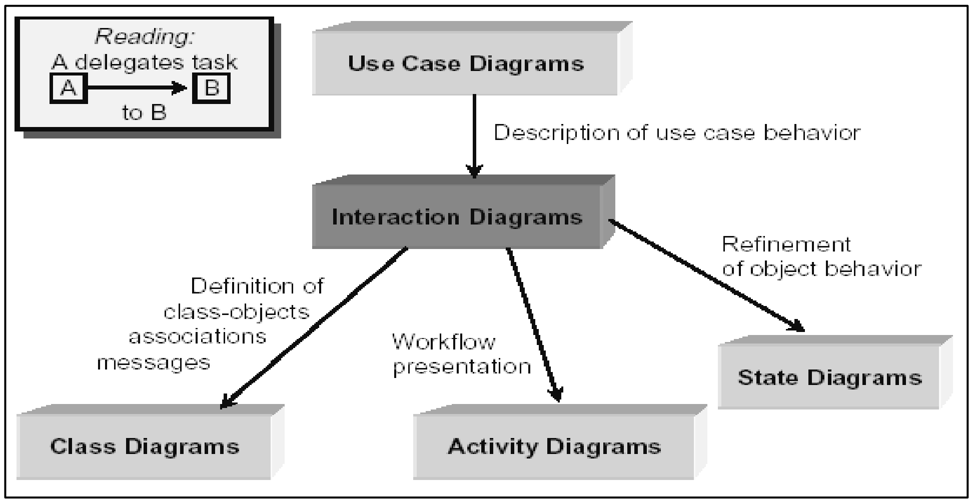


Figure 7 UML Models [BRJ1999]

2.1.1.1 Use Case Model

The functionalities from user's perspective are represented using Use Case Model. It represents functionalities in the form of use cases, actors invoking them and their relationships using an ellipse, a stick figure and connectors respectively.

2.1.1.2 Class Model

Class model represents structure of a system in the form of classes and their relationships. In a class model, classes are represented by boxes with three compartments: name of the class, attributes of the class having name, type and visibility, and methods of the class having name, argument list, return type and visibility. The class relationships can be three types: generalization, association and aggregation. Generalization is represented by line with hollow arrow head pointing towards super class. Association is represented by solid lines between two classes with roles mentioned at the ends. Aggregation is a diamond at aggregate end of the link. The ends of association and aggregation relationships may be annotated with multiplicity giving a range of numbers denoting how many instances of the class at one end can be associated with each instance of the class at the other end. The large rectangles with folded corners are notes where we can write informal clarification.

2.1.1.3 Interaction Model

The dynamic behaviour of the system can be analyzed using behavioral UML models. The interaction model models the interactions that occur between actors and objects in the system in order to carry out the behaviour specified in the scenarios. Interaction Diagrams talk about how groups of objects collaborate in some behaviour. These models represent time ordering of messages. Graphically the objects are arranged on x-axis and messages, ordered in increasing time, along the y-axis.

2.1.1.4 State Model

Response to events varies depending upon the passage of time and the events that occurred already. A model of state behaviour in the state diagram captures all the possible responses of a single object to all the use cases in which it is involved. We have used state models to model different phases through which an agent passes through while executing a request. Graphically the start state and final state can be depicted by a filled circle and bull’s eye respectively. The different states can be shown using rectangles with rounded corners

2.1.1.5 Activity Model

It is a flow chart showing flow of control from activity to activity. One can model sequential and concurrent steps in the modeling. It can also model the flow of an object as it moves from state to state at different points in the flow of control. Activity diagrams can help visualize, specify, construct and document the dynamics of a collection of objects. Graphically the activities can be shown by rectangles with rounded edges and meaningful name, transitions by arrows, conditions by guard, the decision by diamond. One can also show object flow and swim lanes.

2.1.2 Object Constraint Language OCL

Several extensibility mechanisms are part of UML; stereotypes (model elements similar to standard ones but with additional constraints), tagged values (that can be attached to any model element to contain additional information) and constraints (that can be used to create semantic relationship among model elements that specifies Boolean conditions and propositions). Constraints can be expressed using a specific language, Object Constraint Language, OCL.

2.2 UML as Ontology Representation Language

The utility of a visual syntax for modeling languages has been shown in practice and visual modeling paradigms like Entity-Relationship (ER) model or the Unified Modeling Language (UML) are frequently used for conceptual modeling. The UML and OWL relevant for ontology representation have been discussed in this section.

2.2.1 UML Features for Ontology Representation

Package: This is a mechanism for introducing a scope within which one can make logical inferences according to a specified logical formalism. Packages can import other packages, and a package can reference entities in other packages without necessarily importing other packages. This feature is useful while working with multiple ontologies.

Class: This is a set of similar objects. Every class has attributes representing the data it has and methods which are operations the class can perform on the data.

Relationships: This represents relationships among classes. The related classes can collaborate. Three types of relationships like generalization, aggregation, and association are possible.

Constraints: The constraints from the business model can be represented using OCL constraints

Properties: Object properties are represented as UML n-ary associations, while the datatype properties are UML attributes.

Datatypes: Datatypes are represented in the form of a stereotyped UML class.

2.2.2 Ontology Features

Package: Package notion is important for ontology management. Any form of reasoning or inference requires that the scope of the inference be made explicit. By explicitly labeling a package with its logic, one can ensure that any facts inferred by one agent will be the same as those inferred by others.

The Object Constraint Language OCL can be used to express constraints in the UML models. Although there are deficiencies in automated reasoning for UML-OCL based ontology modeling, the reasoning requirements of future multi-agent semantic web based applications are not likely to be predominant design criteria and other issues such as coping with design complexity, interoperability and scalability are more important.

Class: A class is a collection of similar entities.

Property: Ontology Class attributes or associations are represented through properties. Properties represent named binary associations in the modeled knowledge domain. OWL distinguishes two kinds of properties, object properties and datatype properties. The properties can be related using two types of relations. Property subsumption (subPropertyOf) specifies that extension of a property is a subset of related property. Similarly, property equivalence (equivalentProperty) defines extensional equivalence. One important difference between ontology languages and UML is support for first-class properties. UML associations are secondary to the classes that they relate. In ontology languages, one can reference a property without any reference to a class. In this sense property is a standalone entity in ontology languages.

Datatypes: The datatype for OWL is provided by XML schema, which defines a predefined set of named datatypes (PrimitiveType). The users may specify enumerated datatypes (EnumeratedDatatype) which consists of several data value of items (DataValue).

Classifier: represents a concept for grouping resources with similar characteristics.

Constraints: Property can be constrained in various ways. There are global constraints as well as local constraints which are applicable only within a class are possible.

2.2.3 Comparison of Features of Modeling Language and OWL

The semantic web activity group w3c has accepted OWL (Ontology Web Language) as a standard for semantic web ontologies [DS2004]. In spite of many differences, OWL and UML share many similar constructs. Some recent work presents similarities between MOF and RDF [DGD2007]**,** between RDF and Object- Oriented Languages [PU2006]**.**

OWL/ RDF	**Metamodeling Languages**
Ontology	Package
Class, Classifier	Class
Instance and Attribute Values	Individual and Values
Model Element	Resource
Property	Association, Attribute
Data Types	Data Types
Subclass, SubProperty	Generalization Class, Generalization Association
Enumeration	Enumeration
Navigable	Domain, Range
Disjointness, Cover	Disjointness, Union
Multiplicity	Cardinality

Table 1 OWL/RDF and Meta-modeling Languages: Comparable Features [Bhi2009b]

The work about the similarities between software engineering modeling languages and ontology representation languages has been summarized in Table 1. This section discusses these feature comparisons.

2.2.3.1 Class (OWL/RDF) & Class (Meta-modeling Language)

In ontology, a class represents a concept which is abstracted from the actual problem domain. We enumerate all the terms relevant to the problem domain and select those concepts first which exist independent of other classes. Such classes form the anchor in the class hierarchy structure.

When we model a problem domain using Meta-modeling languages like UML, we follow a very similar process. A proper noun analysis is done to form the first list of candidate objects. After going through a well defined process of refinement final list of classes is made.

In ontology development, the class hierarchy can be constructed in three ways:

- A top down development process where one starts with an independent concept and look for specialization of those concepts in our list.
- A bottom up development process where one starts with concepts that are very specific and then generalize them to move up in the hierarchy.
- A combination development process is a combination of the two processes discussed.

The Top-Down process refers to Specialization and the Bottom-Up process refers to the Generalization process of meta-modeling language UML.

2.2.3.2 Properties in OWL & Relationships in UML

In ontology, a class representing a concept is not enough to cover all the aspects of the concept. Hence to describe the internal structure and the relationship it has with other classes we need to define properties.

After taking the classes and sub classes away from our initial list, whatever is left, most of them are properties. There are 3 type of properties supported in OWL:

Object Property

This property covers the relationships between different concepts. A property is supported by

- Domain
- Range

Data Type Property

This property has a class as its domain and its range is some data type like String, Date etc

Annotation Property

This property adds more information about a property or a class.

A property in OWL can be further defined by adding on several characteristics like-

Functional Property: It ensures that only one individual is related with another individual. In case of data type property, one single value is taken.

Transitive Property: a property relating A to B and then B to C is said to be transitive if it can be inferred that A is related to C with the same property.

Inverse: A property linking individual A to B, then the inverse property will relate B to A. Similarly inverse functional properties can be defined.

2.2.3.3 Types of relationships in UML

Association

When two classes are connected to each other in any way, an association relation is established. An association can be unidirectional or bidirectional.

Aggregation

Aggregation refers to *is a part of* relationship.

A class diagram has various classes connected with each other through different type of relationships. The only way to connect different concepts (or classes) in ontology is through Object Type Properties. Apart from Generalization and Specialization, all the relationships are managed by Object Type Properties. Using its characteristics like functional or inverse or transitive one can bring out the association and other relationships. Binary Association is equivalent with an Inverse Object Type Property. A Unidirectional Association is equivalent with the Non Inverse Object Type Property.

2.2.3.4 Objects in UML and Instances in OWL

The instances in OWL are in fact objects in UML. After defining the ontology we instantiate the concepts so that it represents the problem domain we targeted it for. In a Meta-modeling language like UML, once the class diagram is complete, classes are instantiated so as to represent the objects conforming to the defined classes.

2.2.3.5 Package and Ontology

A package provides the ability to group together classes and interfaces that are either similar in nature or related. Grouping these design elements in a package element provides for better readability of class diagrams, especially complex class diagrams. An ontology is a formal explicit description of concepts in a domain of discourse (classes (sometimes called concepts)), properties of each concept describing various features and attributes of the concept (slots (sometimes called roles or properties)), and restrictions on slots (facets (sometimes called role restrictions)). Ontology together with a set of individual instances of classes constitutes a knowledge base.

The biggest obstacle in the way to map UML to ORL like OWL is the notion of Property. The notion of Property in KR languages corresponds to the notion of Association in UML. A Property is a first class element in all ORLs. This means that a Property can exist as an independent element of a language without being attached to Classes. But the UML Associations are connected to classes with association ends. So Association does not have independent existence. It also means that each association is unique. Thus, UML Associations express local restrictions while ORL Property can represent global restrictions. The correction to this problem has been suggested by Baclawaski [B+2001a] in the form of MOF specification which has notions of Property and Restrictions. Both Property and Restriction are modeled as UML Classifiers, enabling them to be a first-class modeling primitive in UML. Also a Property has an aggregation of zero or more Association Ends, so it can exist without Class. The mappings for ORL that we have used [DGD2007] are based on Baclawaski's work.

2.3 Processes of Ontology Building and UML Class Model Building

For Ontology Modeling, there is no one correct way to model a domain— there are always viable alternatives. The best solution almost always depends on the application that you have in mind and the extensions that you anticipate. Ontology development is necessarily an iterative process. Concepts in the ontology should be close to objects (physical or logical) and relationships in your domain of interest. These are most likely to be nouns (objects) or verbs (relationships) in sentences that describe the domain.

Acquiring domain knowledge consists of assembling appropriate information resources and expertise that will define, with consensus and consistency, the terms used formally to describe things in the domain of interest. These definitions must be collected so that they can be expressed in a common language selected for the ontology. Organizing the ontology requires to design the overall conceptual structure of the domain. This will likely involve identifying the domain's principal concrete concepts and their properties, identifying the relationships among the concepts, creating abstract concepts as organizing features, referencing or including supporting ontologies, distinguishing which concepts have instances, and applying other guidelines of your chosen methodology. Then we add concepts, relations, and individuals to the level of detail necessary to satisfy the purposes of the ontology. In the end, the verification of final ontology is done by domain experts.

Building a UML class model requires us to follow object-oriented methodology of developing it. We first gather the domain knowledge and requirements. Analyze those using a method like noun analysis to get list of candidate classes. This candidate class list is then subjected to various criteria checking for vagueness, duplication of these classes. Some more criteria that are used are checking the out of scope of the system classes, possible attributes, associations and roles nouns. We get a truncated list of nouns which is our list of classes. We now try to analyze what each class is supposed to know and we get their attributes. We investigate the possible functionalities of the classes in the form of use cases and their scenarios. When we walk through the scenarios which a set of actions responsible for that use case scenario, we can find out the relationships among classes. For ontology modeling purpose we need to limit ourselves to the list of classes and their hierarchies. So the way we build class models fits well the general ontology building process.

When large and complex digital libraries are to be built, it is very hard even for experts to manually write those XML files without any assistance from the tool. It is also very difficult to understand and comprehend the big picture of a digital library just from a huge set of XML or OWL files. Our work proposes and demonstrates use of software engineering modeling language to model domain knowledge for a digital library. The XMI mapping available from UML models to OWL and Java will allow to create the OWL ontologies for semantic web based digital libraries automatically from the UML domain model.

2.4 Use of UML Models for Agent Behaviour Modeling

Behavioural UML models are useful for expressing interactions among agents. Interaction diagrams capture the structural patterns of interactions among objects. The graphical layout of the sequence diagram emphasizes the chronological sequence of communications. Activity diagrams and statecharts capture the flow of processing in the agent community.

The UML use case model allows us to represent functionalities of agents in the form of use cases. This in turn will help in analyzing the agent interactions better. Semantic web agents will have to carry out the task of finding out and using the information on the web from several resources on their own, processing them and integrating the results and presenting them to the users or carrying out inferencing based on those results. The interaction model helps in analyzing the interactions among agents as they take part in query execution. The message passing among the agents can be represented as a function of time. The agents and objects taking part in interactions can also be shown in interaction models. The various constraints can be represented using OCL elements.

2.5 Summary

The software engineering modeling technique UML is based on MDA. The UML models are classified as structural models and behavioural models. UML Class Model is a structural model which can be used to model the problem domain. The OCL can be used to model constraints. Features of UML are analyzed for a possible use of UML as a domain modeling language for semantic web based systems. Equivalent OWL features are discussed. The UML features relevant for semantic web ontology representation have been mapped onto the corresponding OWL features. This will allow us to make use of UML class model to represent domain ontologies for any semantic web based system. The Use Case, Interaction, Activity, State models are behavioural. Use Case model models functionalities of software. Interaction model can be used to model interactions and message passing among objects of classes participating in processes. Activity model can analyze the flow of processes for any software and State model models state changes of objects in response to various events. Each of these models can be used to analyze and design different behavioural aspect of any semantic web based system.

3. Research Methodology for Semantic Web based Systems Modeling

XSLT mappings based on the transformation established in chapter 2 are developed for mapping from UML models to Java and OWL representations. The research methodology used to map UML models to Java and OWL representations have been discussed in this chapter. The XSLT technique used by this methodology is discussed in section 1. The UML models are developed using rational rose tool which is described in section 2. Section 3 contains Code generation from the UML models using XSLT technology. It is based on the fact that UML models have XMI representations.

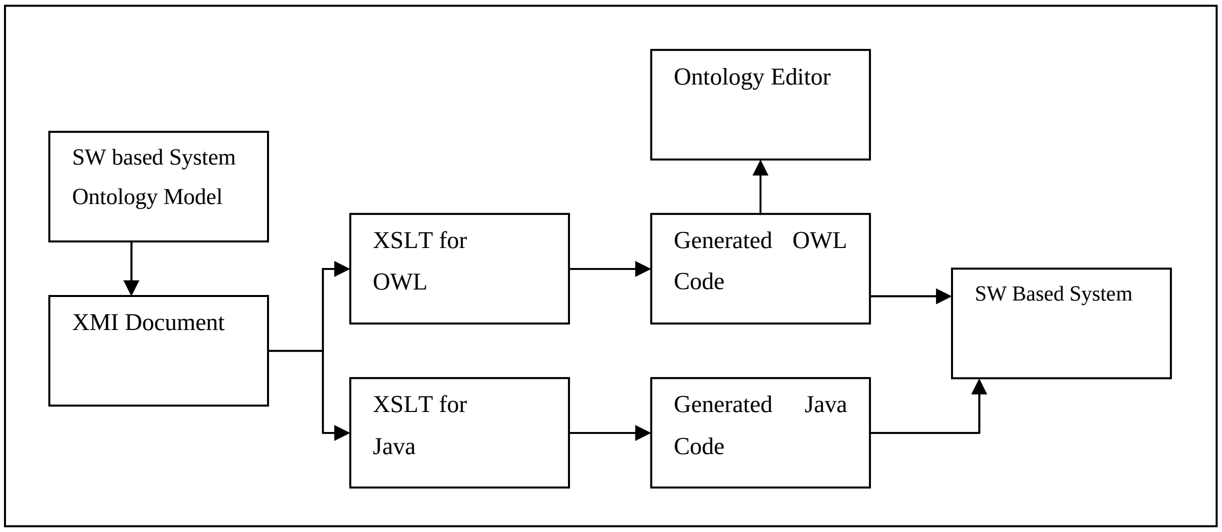


Figure 8 Schematic of UML to ORL and Java Mapping [Bhi2008]

Our approach separates the design work for the semantic web based systems from the technical details of actual implementation. We can achieve this using software engineering modeling techniques; MOF based MDA and UML as described in the previous section. The chosen visual model, Class model will show the structure and different concepts of a SW based system and the relationship among these concepts. The suitability of class model for ontology representation has already been discussed in the section on UML as Ontology representation language. The MDA and UML approach allows us to make use and reuse of models directly without bothering about implementation details. The technology changes very fast, but the models created will be valid even then and only implementation of the models using new technology needs to be done again. This saves lots of resources. The XML Model Interchange Language defines a standard way to serialize the UML diagrams. UML classes can also be mapped to sets of Java classes and RDF [DGD2007].

The domain experts create an ontology using ontology editor and the graphical representation of ontology using UML tool like Rational Rose. The UML is a graphical language hence very easy for human comprehension. Hence the domain experts are expected to contribute substantially for the development of the domain ontologies using UML tools. The XMI files are created as shown in Figure 8. A pair of eXtensible Stylesheet Language Transformation (XSLT) then creates Java files and ORL representation of ontology respectively. The Java classes can be used by the applications for representing knowledge as in memory data structures. The ORL representation can be used for domain specific information. Presently the RDF transformation of XMI files is available and a very little work has been done for direct XMI representation of UML model mapping to OWL [DGD2007]. As shown in Figure 8, the following processes take place:

- UML models are exported using XMI Extensible Language Model Interchange format
- The XMI format serves as input to the transformation process as do the XSLT transformation rules
- Run XSLT for Java and OWL transformations
- Java and OWL specification can be obtained which can be understood and used by semantic web intelligent agents directly.

3.1 Transformation using XSLT

Both UML and OWL can be serialized in XMI format so that we can do the transformation from UML to OWL using XSLT. Since XMI and OWL use XML syntax, we can convert the structure and content of a XMI document to OWL document through

XSLT. A transformation expressed in XSLT describes rules for transforming a source tree into a result tree. The transformation is achieved by associating patterns with templates. A pattern is matched against elements in the source tree. A template is instantiated to create part of the result tree. While transforming if XSLT finds predefined pattern in the XML documents, it replaces the pattern with another according to the rules. This thesis uses the templates defined in work by [DGD2005] which are based on Baclawaski's transformations [B+2001b].

3.2 UML and Ontology Tools

This subsection describes UML tool Rational Rose that was used to model ontologies for semantic web based system. The ontology editor tool Protégé has been used to for demonstrating comparable OWL features.

3.2.1 Rational Rose Tool

ROSE Rational Object Oriented Software Engineering uses UML to provide graphical methods for non-programmers wanting to model business processes as well as programmers modeling application logic. Rose has divided the entire model into three views; Use Case View, Logical View, and Component View [BRJ1999]. Use Case View consists of Use Case Diagrams. Use Case Diagrams depicts Use Cases, actors and the association between them. The rational rose interface contains components; standard toolbar, diagram toolbox, browser diagram window, and documentation window as shown in Figure 9.

Standard toolbar

It is arranged across the top of the screen. Placing the cursor on an icon displays the tool tip for that icon.

Diagram toolbox

It is placed between the browser and the diagram window. The diagram toolbox changes based on the active diagram. Placing the cursor on an icon displays the tool tip for that icon.

Browser

It is placed at the left of the screen. The browser is a navigational tool to display the names and icons representing diagrams and model elements. Double-clicking a diagram icon, this is displayed in the diagram window. Clicking a model element the specification window for this element is displayed.

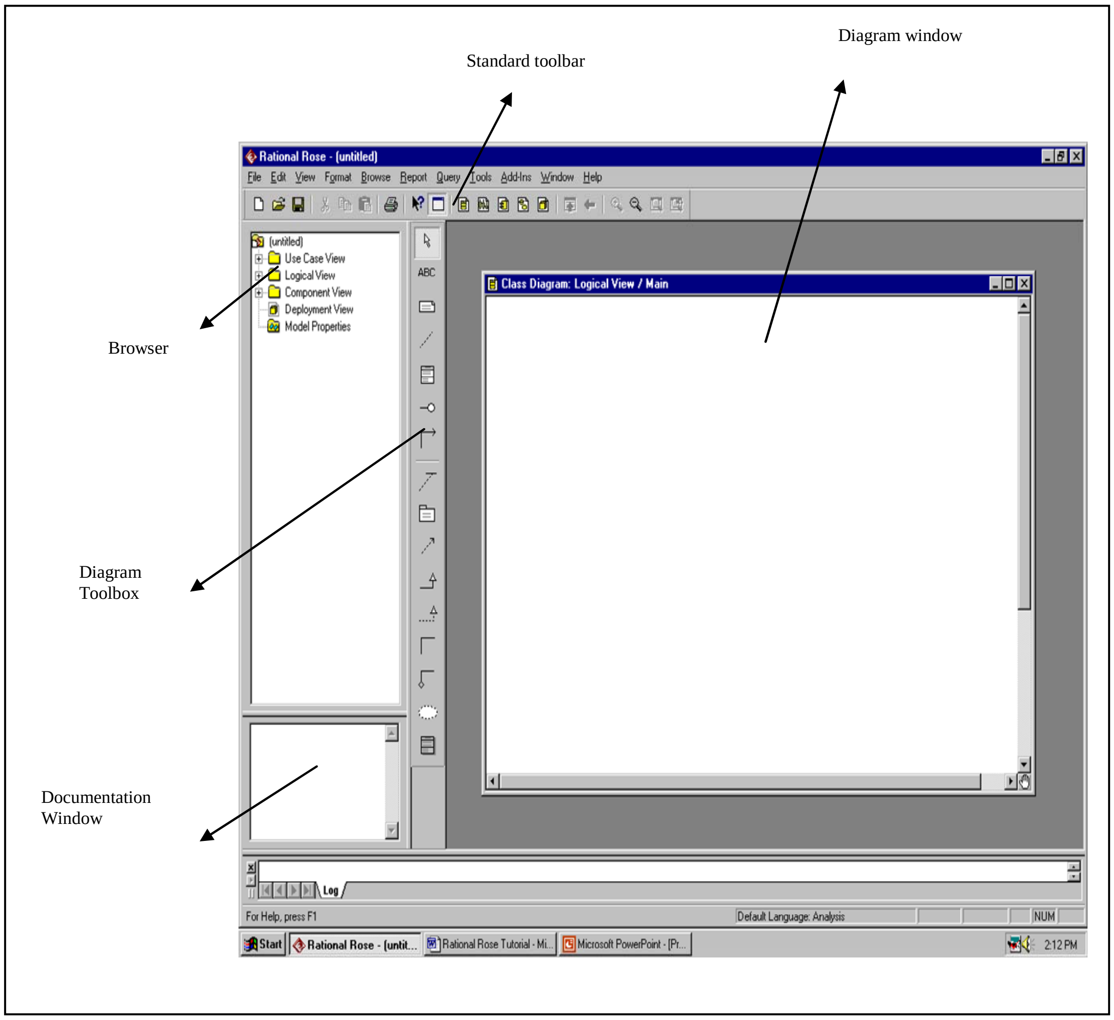


Diagram window
Standard toolbar
Browser
Diagram Toolbox
Documentation Window
Rational Rose - (untitled)
File Edit View Format Browse Report Query Tools Add-Ins Window Help
(untitled)
Use Case View
Logical View
Component View
Deployment View
Model Properties
Class Diagram: Logical View / Main
Log
For Help, press F1
Default Language: Analysis
NUM
Start
Rational Rose - (untit...
Rational Rose Tutorial - Mi...
Microsoft PowerPoint - [Pr...
2:12 PM

Figure 9 Rational Rose Tool Window View

Diagram window

The diagram window is used to create, display or modify Rose diagrams in the diagram window.

Documentation window

The documentation window displays the information about the diagrams and models elements. This information can be created, viewed or modified in this window or in the documentation window of the specification.

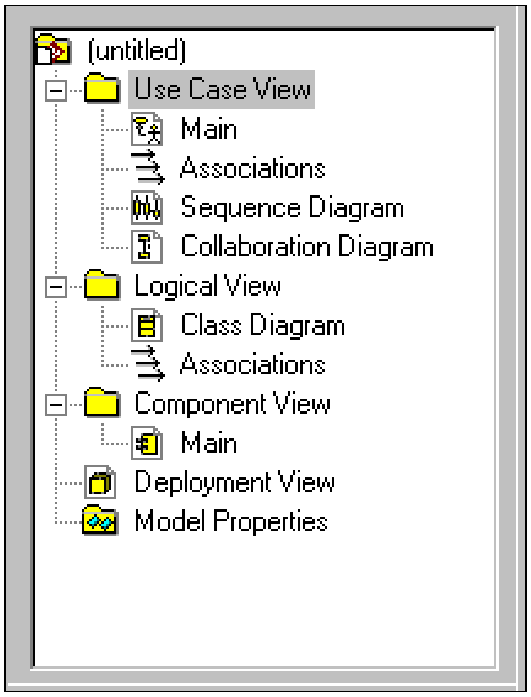


Figure 10 Typical Rational Rose Browser View

Views

Rational Rose is organized around different views of a software project as depicted in Figure 10. Each view represents a different aspect of the model. Each view has associated different diagrams. In the browser are showed the different software views and the type of diagrams.

- Use Case view: Use-case diagrams, sequence diagrams, collaboration diagrams and activity diagrams.

- Logical view: Class diagrams and state chart diagrams.
- Component view: Component diagram.
- Deployment view: Deployment diagram.

3.2.2 Protégé Ontology Editor

Ontology editors facilitate development and management of ontologies, the definition and modification of concepts, properties, axioms, and restrictions. Common to most ontology editors is the ability to create a hierarchy of concepts and to model relationships between those concepts.

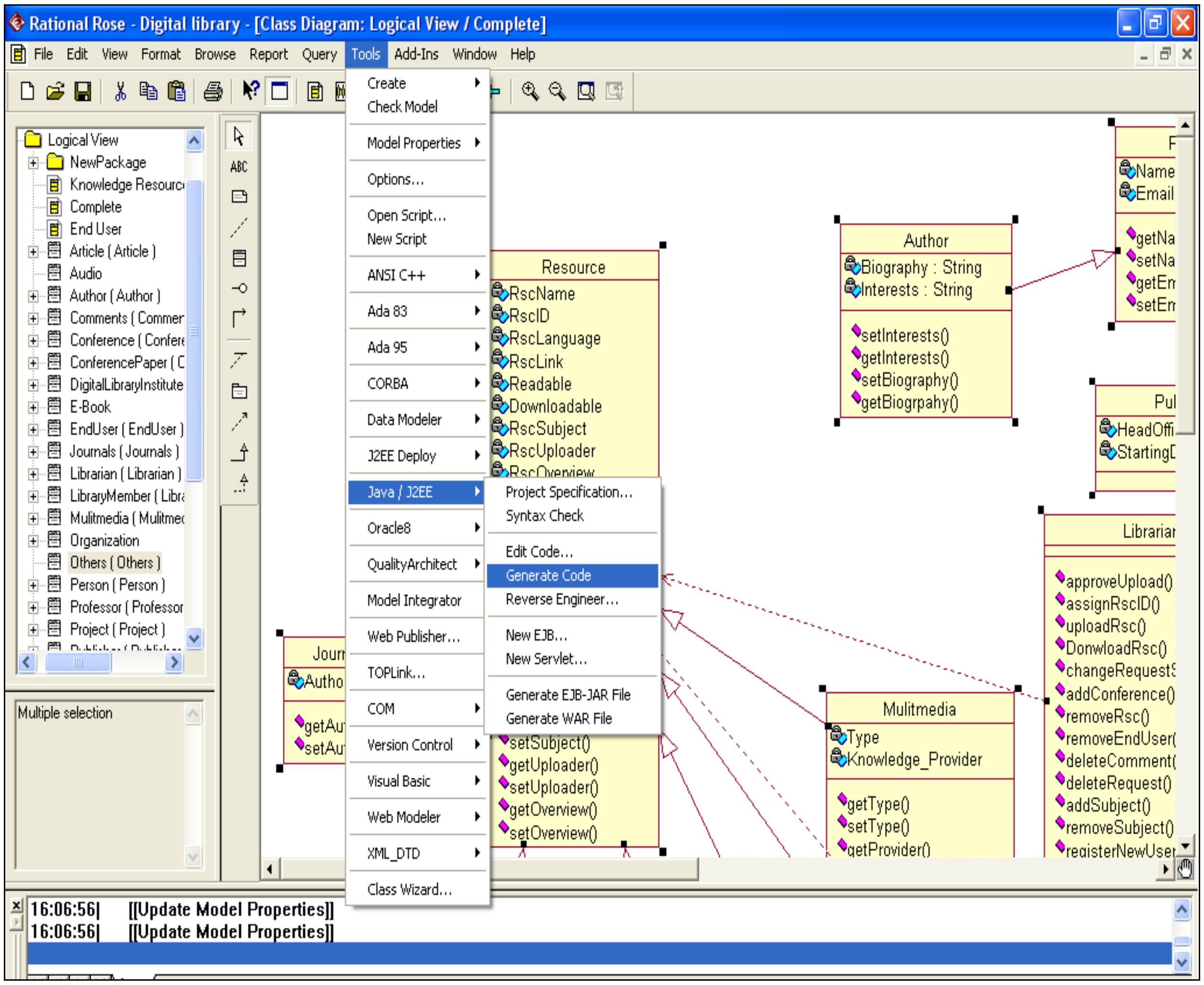
Protégé [NM2000] is an open source ontology tool. It allows user to construct domain ontology. It offers an interface which allows to model ontologies. Protégé has been used for demonstrating the OWL features for the equivalent UML features depicted using rational rose tool in this thesis work.

3.3 Code Generation from UML Models

Using the XSLT transformations, one can generate Java and OWL codes from the UML class model directly. The process of generating the code from UML class models has already been depicted in Figure 8.

3.3.1 Generating Java code from UML using Rational Rose

The UML class model has been used to generate the Java skeleton code. The UML classes and interfaces map to their equivalent Java classes and interfaces. The UML associations map to Java fields. The Java mapping facility available in rational rose tool which has in built implementations of the transformations proposed by Cranefield [Cra2001a] has been used**.** This transformation assumes that UML model uses the OCL primitive types Boolean, Integer, Real, and String. These are mapped to the corresponding Java class types.



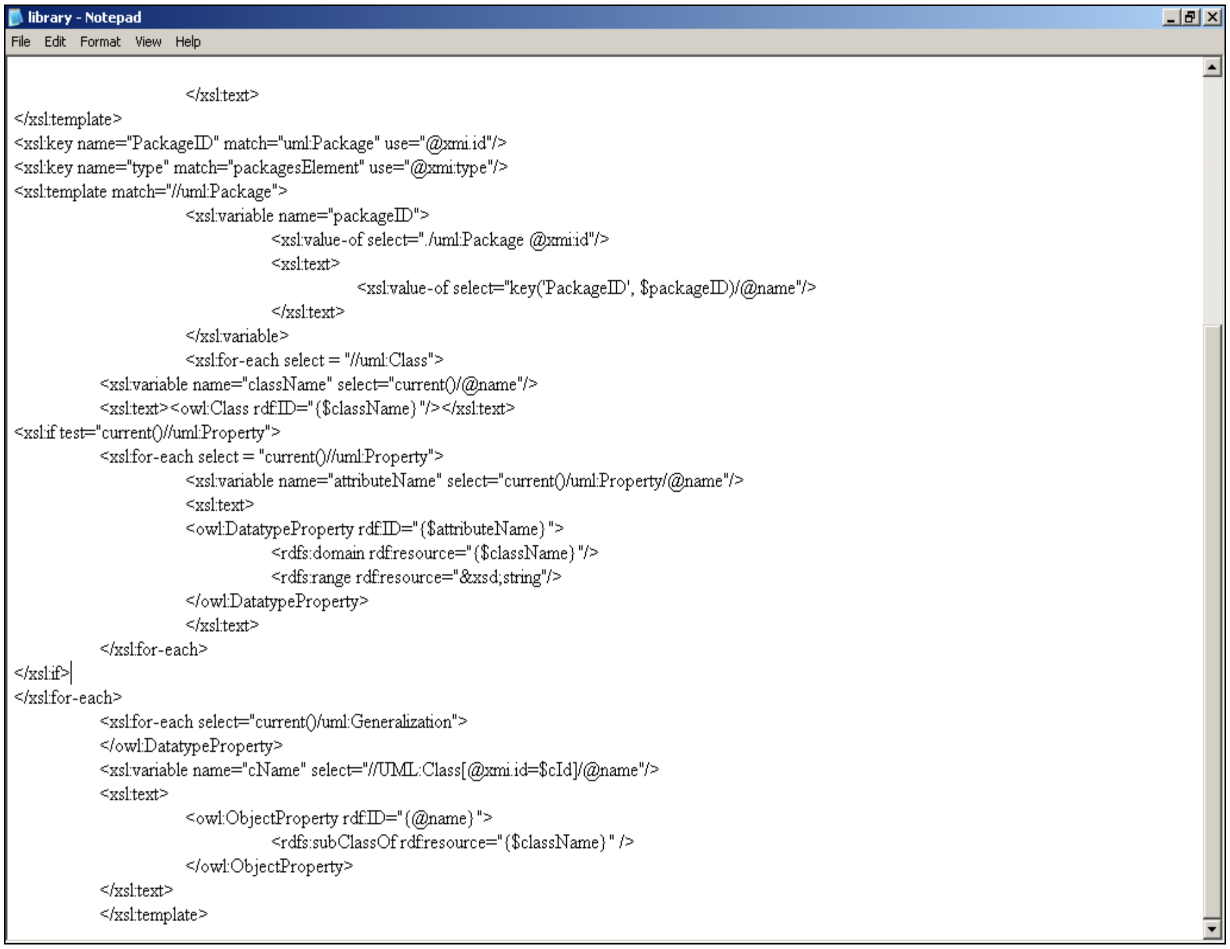
```
16:06:56|     [[Update Model Properties]]
16:06:56|     [[Update Model Properties]]
```

Figure 11 Java Code Generation from Rational Rose

This section describes the skeleton Java code generation from UML diagrams using rational rose tool which is depicted in Figure 11. It includes:

- Creating Class Model includes creating class diagram, entering all the information about attributes such as their type (String, Boolean etc), and entering return types of all the methods.
- Generation of Code (as shown in Figure XXX) includes selection of all the classes by Ctrl + A and Go to Tools >>> Java/J2EE>>>Generate Code.
- Entering the class path includes clicking on Select All and then Clicking OK.

- Finally one has to look for errors in the Log. The rational rose has its own code editor. Right click on the respective class in Logical View and select Java J2EE / Edit Code to edit the corresponding code.



```
        </xsl:text>
</xsl:template>
<xsl:key name="PackageID" match="uml:Package" use="@xmi.id"/>
<xsl:key name="type" match="packagesElement" use="@xmi:type"/>
<xsl:template match="//uml:Package">
        <xsl:variable name="packageID">
            <xsl:value-of select="./uml:Package @xmi:id"/>
            <xsl:text>
                <xsl:value-of select="key('PackageID', $packageID)/@name"/>
            </xsl:text>
        </xsl:variable>
        <xsl:for-each select = "//uml:Class">
    <xsl:variable name="className" select="current()/@name"/>
    <xsl:text><owl:Class rdf:ID="{$className}"/></xsl:text>
<xsl:if test="current()//uml:Property">
    <xsl:for-each select = "current()//uml:Property">
        <xsl:variable name="attributeName" select="current()/uml:Property/@name"/>
        <xsl:text>
        <owl:DatatypeProperty rdf:ID="{$attributeName}">
            <rdfs:domain rdf:resource="{$className}"/>
            <rdfs:range rdf:resource="&xsd;string"/>
        </owl:DatatypeProperty>
        </xsl:text>
    </xsl:for-each>
</xsl:if>
</xsl:for-each>
    <xsl:for-each select="current()/uml:Generalization">
    </owl:DatatypeProperty>
    <xsl:variable name="cName" select="//UML:Class[@xmi.id=$cId]/@name"/>
    <xsl:text>
        <owl:ObjectProperty rdf:ID="{@name}">
            <rdfs:subClassOf rdf:resource="{$className}" />
        </owl:ObjectProperty>
    </xsl:text>
    </xsl:template>
```

Figure 12 Snapshot of XSLT for Transformation from UML to OWL [Bhi2009a]

3.3.2 Generating OWL Code from UML Model

The OWL code has been generated from the UML class model using the transformations defined in Djuric’s work. [DGD2005]. The UML models can be exported to XMI format. The transformations exploit the fact that the both XMI and OWL are encoded using XML. UML models can be exported to XMI format. A snapshot of XSLT transformation based UML to OWL mapping has been shown in Figure 12. The UML models are exported to XMI format. The XMI format serves as input to the transformation process. The transformation rules have been defined in XSLT processor presented by Djuric. The structure and contents of a XMI document can be converted to OWL through a XSL language. The XSL uses templates. It looks for predefined pattern in the XMI documents, replaces the pattern with another one according to the XSLT rules.

3.4 Summary

The software engineering modeling technique UML has been proposed to be used for modeling different structural and behavioural aspects of a semantic web based system. XSLT mappings help in mapping from XMI representations to Java or OWL code which is directly understandable by semantic web applications. The Java and OWL XSLT mappings can transform XMI representations to the corresponding Java and OWL code. The approach is based on a simple fact that the UML models have XMI representations. This can facilitate reuse of existing models which have XMI representations. This will further help to transform models to code directly which is understandable by agents.

4. Digital Library Domain Modeling Using UML

The semantic web based systems allow us to offer semantics based services. These systems are ontology based and for such systems building good ontologies is very important. Building good ontologies needs heavy involvement from domain experts but lack of good ontology tools with visual interfaces has always been a problem. To overcome this problem, this thesis suggests use of well known software engineering modeling technique, UML and MDA to build ontologies for semantic web based system. This work also includes use of other behavioural UML diagrams to model semantic agent behaviour. Digital library is a managed collection of digital objects and services associated with the storage, discovery, retrieval, and presentation of those objects which provides digital knowledge resources for universities, organizations and individuals. With the rapid growth of information, knowledge resources, and user demands, the problem of how to organize and retrieve the complex information has become an important issue for digital library research domain. The ontologies can be used to express explicitly the semantics of structured and semi-structured information in order to support information acquiring, maintaining and accessing in automated fashion. Digital library agent behaviour has been analyzed using various behavioural UML models.

The research methodology that has been used for ontology building and agent modeling is described in Chapter 3. In this chapter, Chapter 4, the domain model has been built for digital library case study. The digital library requirements and domain model have been developed and presented using rational rose tool.

4.1 Digital Library Requirements

A library is seen as a large pool of informative resources. From the last decade, significant changes have occurred in the way the resources have emerged. The World Wide Web has changed the entire scenario. Internet has emerged as the largest library of information covering every subject and knowledge area. Therefore a transition from normal library to Digital Library is essential. A digital library is a collection of resources; books, journals, conference papers, audio tapes, video files etc.

Digital libraries collect, preserve and provide the resources in digital format. Digital Libraries offer access to large amounts of content in the form of digital documents. Many of them have evolved from traditional libraries and concentrated on making their information sources available to a wider audience by scanning journals and books. The present digital libraries are large repositories of e-books, journals, magazines, articles, conference papers and other multimedia resources in the form of audio and video files. As the size of such digital library increases, the problems with maintenance and resource retrieval increases. An efficient storage and proper organization of resources is must for a digital library. Having large collection does not help the user community until and unless it is easily accessible and has sharp search facilities. As the size of such database increases the search result set also increases, making it more difficult for the user to find a desired resource. For the librarian also, the management and organization of large resource increases. Most of the digital libraries are working as independent entities. If such digital libraries can have same structure for organization of resources and work in cooperation with each other, the user community will gain a lot. This will eventually lead to a federation of digital libraries. The federation of digital libraries will have various advantages over its normal counterpart. The large collection of resources can allow uploads from both librarians as well as users. The federation also provides for better management of resources as the individual entities will be working in cooperation with each other. The interactions among federated library user communities have been expected to increase substantially.

Existing digital libraries are large databases of resources. A good number of services are provided like search, bookmark etc but the user interactions is one area which still needs development as far as digital libraries are concerned. A federation of digital library system which can simultaneously have resource management with increased and quality user interaction will definitely add value to the system. Users from different universities, countries can post their comments on a resource. A lot of project work goes around in different universities; students can upload their projects and find users with similar goals and can work collectively. Another desired feature can be managing user interests. The intended user of this system can be any person interested in accessing the kind of resources available in the digital library. The user can be a student or faculty having access to the shared content of the library. Any user is supposed to be a member of the system and adhere to the rules and regulations of the system. The administrator will manage the entire system and will act as a moderator. The basic functionalities of such system are; procuring resources, managing resources, giving access to resources and managing different categories of users like administrator and member. Procuring resources needs to search for appropriate resources, get access to these resources through purchase, loan from other libraries, generate. Managing the resources consists of add resource, modify resource and delete the resource. Managing the users consists of functionalities like login, create new member, drop member, manage subscription accounts etc. The authors submit their contributions to the library. The administrator of the library then verifies and accepts or rejects it. The different categories of users expect different set of functionalities from the system.

4.2 Digital Library Functionality Modeling

This section contains modeling of digital library functionalities. Based on the requirements for the semantic web based digital library, functionalities are identified. These are modeled using UML use case model. The different user categories identified for the system have different privileges.

4.2.1 User Classification

The user can be categorized as Student, Faculty and others, and referred as Library members. The Administrator referred as Librarian.

The different user groups use different set of functionalities of the system. These are categorized broadly as follows.

4.2.1.1 Library Members

The end users are the students, professors and others who will be using the digital library. They will use following functionalities:

- Searching of resources, other library members with similar interests
- Downloading of resources
- Uploading of resources
- Adding comments and making requests for resources

Each library member will create an account and access the system through the account.

4.2.1.2 Librarian

The administrator will be the central authority managing the digital library. Librarian will be looking after following aspects of digital library:

- Resource development
- Resource classification
- Managing user accounts
- Adding, deleting, editing, updating and monitoring of resources

4.2.2 Digital Library Use Cases

The digital library will have the functionalities listed below:

4.2.2.1 Collection of Digital resources

Description: The basic functionality of a digital library is to provide its users a large collection of digital resources. The collection of resources can be done in two ways:

- Employing a collection analyst who will be regularly updating the collection or
- A dynamic collection where user can upload various digital resources which will be verified by the administrator before adding it to the collection.

Actors are librarian and professor.

4.2.2.2 Classification of Digital resources

Description: Having a large collection of digital resource will not serve the purpose until and unless these resources are organized in an efficient manner. Hence these resources will be available to the user in a neatly classified interface. Classification can be done on various parameters like subjects for e.g. Science, Philosophy, and Mathematics etc.

Type of resources

- Journals
- Magazines
- Reports
- Various other publications

Actor is the librarian.

4.2.2.3 Individual user account

Description: To make the library similar to its on paper counterpart, it must have membership facilities available. Hence every user will be having its own account though which he can operate in and use the various features of digital library.

Actors: End user who will be creating the account and the Administrator who will be confirming the creation of the document.

4.2.2.4 Uploading of resources

Description: Dynamic resource development will be effective only when every user would have uploading rights. Each user can upload the resources and share the same with other members of the library

Uploading apart from the end users will also be done by administrator of the system.

Actors: Administrator and Library Member

4.2.2.5 Downloading of resources

Description: Similar to uploading, the library members will be allowed to download the various resources present in library

Actor: Library Member

4.2.2.6 Search

Description: One of the most important features for a digital library is an efficient search. The library should have search facilities so that user can easily locate the resource of their desire from the library. Depending on the amount of resources and the various areas that they cover, the search will be provided. Parameters on which search can be done are:

- Type of digital resource
- Author
- Publication
- Publication date

Actors: End user who will be entering the search parameter.

4.2.2.7 Deleting the resources

Description: From time to time, deletion of resources is also important. This will be done by the administrator who will be controlling the resources. He will be responsible for deleting those resources which have wrong information or the information is outdated.

Actors: Administrator and the end user in case if he/she has added the particular resource.

4.2.2.8 Comment facility

Description: The library member can discuss their view points about any resource though comments.

Actor: Library Member

4.2.2.9 Bookmark Facility

Description: The user can also bookmark a particular resource and save its link to his favorites so that he need not search for the same article or resource again and again.

Actors: End user who is having an account.

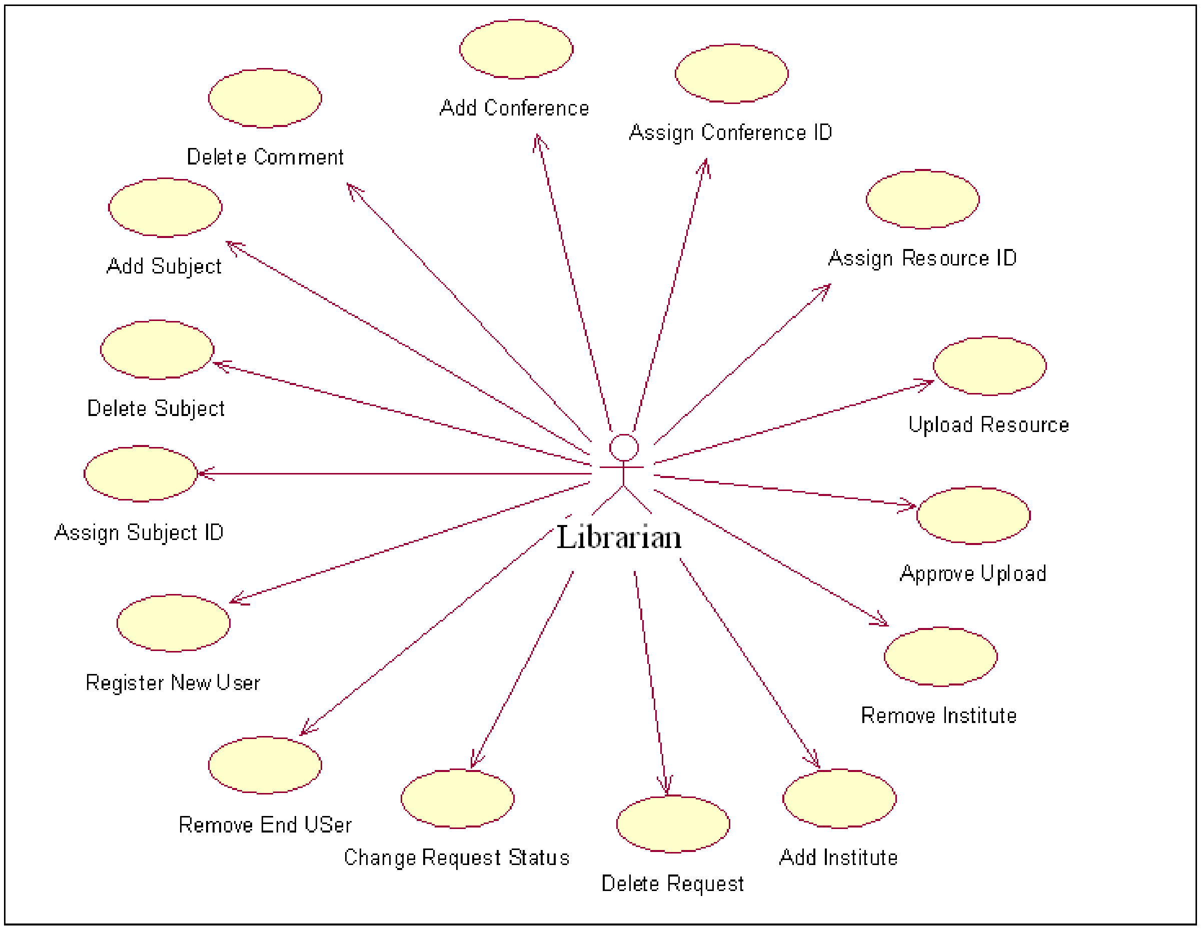


Figure 13 Use Case Model for Librarian using Rational Rose Tool

4.2.2.10 Request Facility

Description: A library member can request a particular resource from the administrator. This will also give a sort of feedback to the librarian.

Actor: Library Member

4.2.2.11 Search among the library members with Similar Interests

Description: A library can become more dynamic and responding if its users can communicate among themselves. Therefore this digital library will manage user interest and will help them in collective learning.

Actor: Library Member

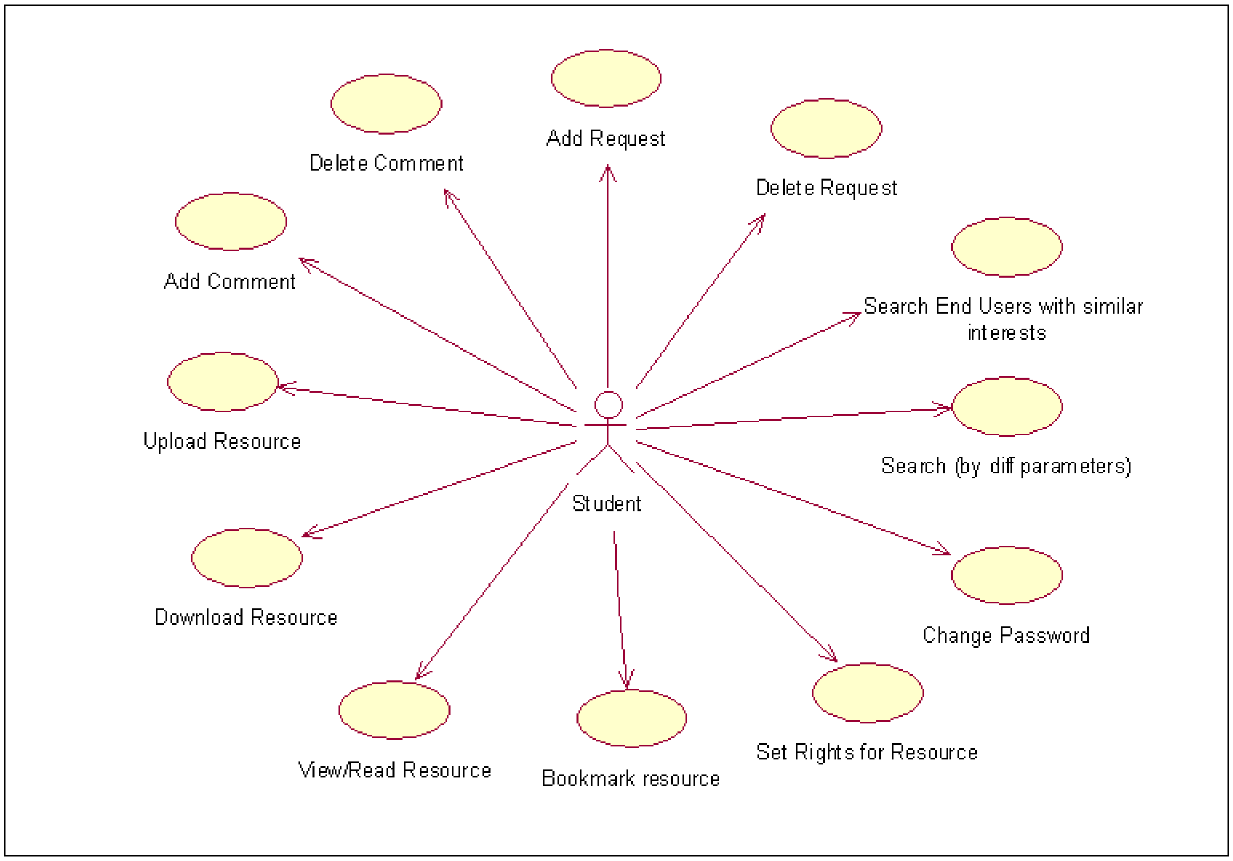


Figure 14 Use Case Model for Student using Rational Rose Tool

4.2.3 Digital Library Use Case Models

The functionalities for these various categories of users have been identified and modeled using the UML use case model. The use case models for the librarian, student, and professor user categories have been developed using Rational Rose and presented in Figures 13, 14, and 15.

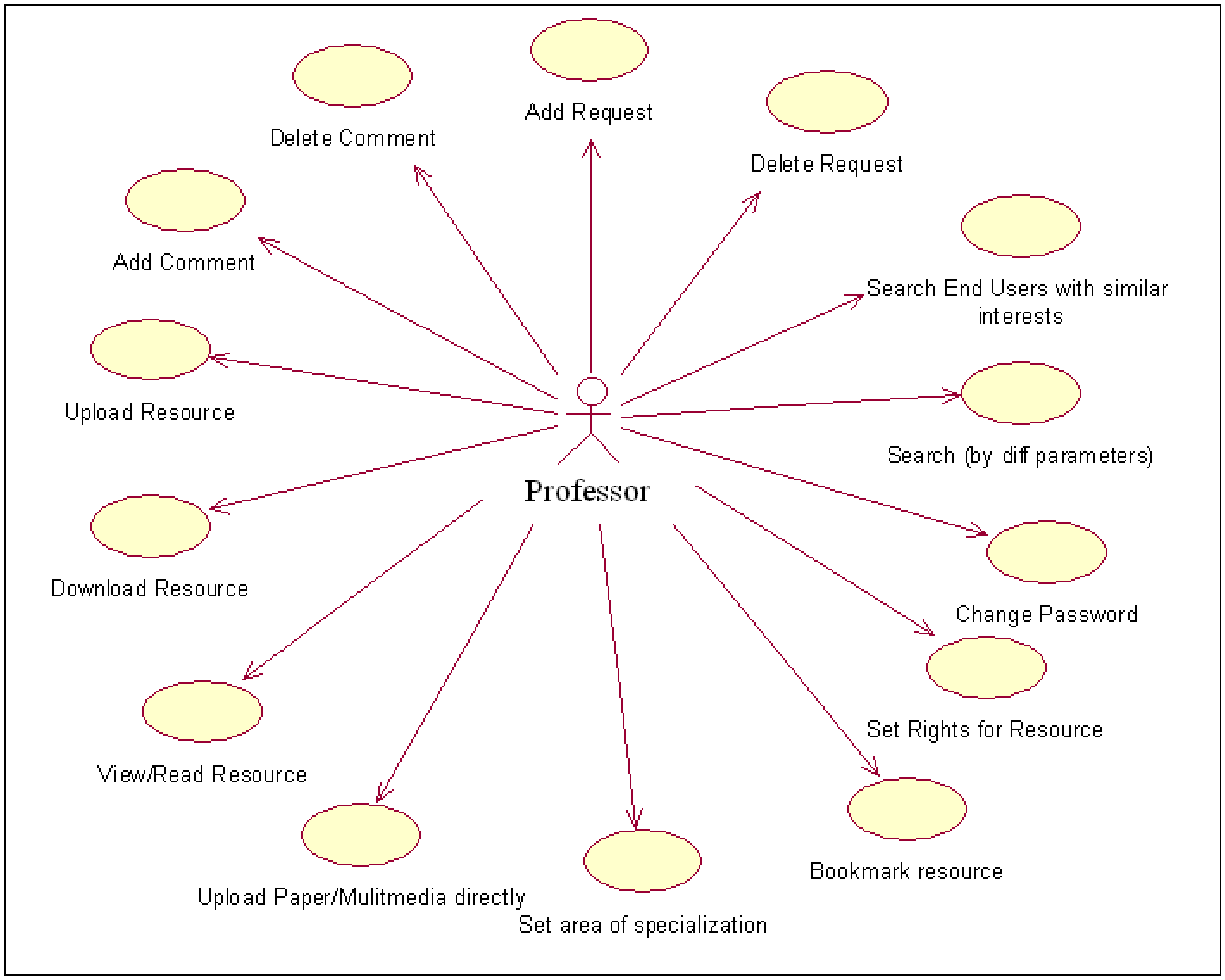


Figure 15 Use Case Model for Professor using Rational Rose Tool

4.3 Digital Library Domain Modeling

The domain knowledge for any semantic web based system can be represented in the form of domain ontologies. Ontology classes represent concepts from the problem domain, relationships like hierarchies, aggregation and association, attributes, constraints and functionalities. A class diagram gives static view of the structure of the system. A class diagram presents set of classes, interfaces and collaborations and their relationships. The class diagrams are often used to model vocabulary and collaborations of the system. The UML class, association and attribute can be used to represent OWL class and properties. The noun and verb analysis from object oriented paradigm domain have been used to find the classes, attributes and methods.

4.3.1 Identifying Concepts Using Noun Analysis

The important concepts from the domain of interest can be identified by noun analysis of problem description. The candidate classes list is obtained from the initial noun list. It has been truncated using various criteria like vagueness of classes, duplicate and irrelevant classes. The candidate classes representing associations, attributes and roles are also removed.

The original list of nouns has been listed below:

Library
Collection
Knowledge
Information
Resource
Book
Documents
Internet
World Wide Web
Subject
Digital Library
Journals
E-book
Magazines
Articles

Conference Papers

Multimedia

Audio Files

Video Files

User

Librarian

Management

Organization

Universities

Countries

Project

Faculty

Administrator

Moderator

The criteria used for trimming this original list of nouns have been listed below:

Duplicates*:* if two or more objects are simply names for the same thing, then only one of these should be used as the basis for a class Example: Librarian is retained and the duplicate administrator is dropped.

Irrelevant: Objects which exist in the problem domain but which are not part of the intended system should also be discarded.

Vagueness: When considering words carefully it sometimes becomes clear that they do not have a precise meaning and cannot be a basis for useful class in the system. Example: Collection or Documents are vague words for the given problem domain.

General: Some words are too general to be accepted as a class. Example: Knowledge is too general as a class.

Attributes: The words which will be kept as information of some class and not a class itself. Only operations identified can be set and get the data. Example: Video file is a type attribute of the resource class rather than a class itself.

Associations: some words actually represent a relationship between objects rather than the class itself.

Roles: The same class may play different roles to fulfill different responsibilities. Example: Librarian and Administrator.

The following final list of nouns forms the list of important classes from the problem domain:

Resource
Subject
Digital library
Journal
E-book
Magazine
Article
Conference Paper
User
Librarian
University
Organization
Project
Faculty

From the domain knowledge, one can easily figure out that *Resource* class will be the super class and following sources of information will become specializations related to it.

E-Book
Journal
Conference Paper
Project

Multimedia

Article

Similarly the *Person* and *End User* classes form the superclass for the related hierarchies.

The End User subcategories are:

Student

Professor

Others

Having previous knowledge about how a digital library works, knowing the actors and which attributes to maintain, The classes and their subclasses have been listed below:

Resource

E-Book

Journal

Multimedia

Conference Paper

Project

Article

Report

Person

Author

Publisher

Library Member

Librarian

End User

Student

Faculty

Others

Organization
Digital Library
Conference
Subject

To increase user-user and user-librarian interaction, 2 classes are added:

Request
Comment

End user can request a resource from librarian. End user can also comment on various resources.

4.3.2 Identifying Attributes, Operations and Relationships

Each class has set of attributes and operations. The attributes are the essential description of a class. They are common structure of what a member of the class should know. The requirements or the description of the problem domain is a useful source for identifying attributes. The operations affect instances of classes. The verb analysis of problem description results in list of operations. An operation can be thought of as a small contribution of one class in achieving larger task represented by a whole use case. The description of use cases helps in identifying the operations. The overall responsibilities of the classes help in assigning various operations to classes. By walking through each use case one can make sure that the identified classes are capable of handling it. Some of the unfulfilled requirements or functionalities may lead to refinement of the classes, operations and attributes. The classes or concepts are related to each other. The UML allows three categories of relations among classes; inheritance, aggregation and association. By going through the use cases and associated scenarios, set of classes that work together closely to carry out some behaviour while realizing some use case have been identified. That further helps to identify the relationships among these classes. The figure shows domain model in the form of class diagram for the digital library that has been created using this method and represented using rational rose tool.

The classes with their attributes and methods have been listed below:

Resource

This class represents the content of digital library. All the information is represented by this class.

Attributes:

- RscName
- RscID
- RscLanguage – Language of the resource
- RscLink
- RscUploader
- RscOverview
- RscSubject
- Downloadable
- Readable

Methods:

- getName()
- setName()
- getLanguage()
- setLanguage()
- getLink()
- setLink()
- getComments()
- readability()
- downloadable()

- getSubject()
- setSubject()
- getUploader
- setUploader
- getOverview()
- setOverview()

Relationships with other classes:

- Librarian
- Library Member
- Subject
- Comment

The **Resource** class has following sub-classes:

Journal

Attributes:

- Author

Methods:

- getAuthor()
- setAuthor()

Relationships with other classes:

- Author
- Publisher

Ebook

Attributes:

- Author
- Publisher

- Edition

Methods:

- getAuthor()
- setAuthor()
- getPublisher()
- setPublisher()
- getEdition()
- setEdition()

Relationships with other classes:

- Author
- Publisher

Conference Paper

Attributes:

- Editor
- ConfID
- Authors
- Organization

Methods:

- getEditor()
- setEditor()
- getConfID()
- setConfID()
- getAuthor()
- setAuthor()
- getOrginzation()
- setOrganization()
- getPapers_sameAuthor()

Relationships with other classes:

- Conference
- Author
- Others

Project

Attributes:

- Developers
- DeveloperEmailID

Methods:

- getDeveloper()
- setDeveloper()
- contactDeveloper()

Article

Attributes:

- Author
- Publisher

Methods:

- getAuthor()
- setAuthor()
- getPublisher()
- setPuvlisher()

Relationship with other classes:

- Author
- Publisher

Report

Attributes:

- Date

- Author
- Organization

Methods:

- getDate()
- setDate()
- getAuthor()
- setAuthor()
- getOrganiztion()
- setOrganization()

Relationship with other classes:

- Author
- Organization

Multimedia

Attributes:

- Type
- Knowledge_provider
- Duration

Methods:

- getType
- setType
- getDuration
- setDuration
- getKnowledgeProvider
- setKnowledgeProvider

Person

This represents any person.

Attributes:

- Name
- Email ID

Methods:

- getName()
- setName()
- getEmailID()
- setEmailID()

This has following subclasses:

Author

Attributes:

- Biography
- Interests

Methods

- setInterest()
- getInterest()
- getBiography()
- setBiography()

Publisher

This represents the Publisher. It can also be a publication house.

Attributes:

- HeadOffice
- Starting date

Methods:

- getLocation()
- setLocation
- getDate
- setDate

Library Member

This represents all the persons which are members of the digital library.

Attributes:

- ID
- Password

Methods:

- getID()
- setID()
- getPassword()
- setPassword()

Since a library has different type of members, this class has been again has following subclasses:

Librarian

Methods:

- approveUpload()
- assignRscID()
- uploadRsc()
- DownloadRsc()
- changeRequestStatus()
- addConference()
- removeRsc()

- removeLibraryMember()
- deleteComment()
- deleteRequest()
- addSubject()
- removeSubject()
- registerNewMember()
- assignInstituteID()
- assignSubjectID()
- addOrganization()
- removeOrganization()

Relationships with other classes:

- Subject
- Organization
- Library Member
- Resource
- Request
- Comment
- Digital_ Library_Institute

End User

End user is one who is the actual user of digital library.

Attributes:

- Link

Methods:

- Search() – Search can be of various types using the parameters. Parameters could be ID, up loader, Subject, Upload date, Interest, Author, conference, subclass of resource.
- readRsc()

- DownloadRsc()
- bookmarkRscLink()
- requestRsc()
- change_password()
- addComment()
- removeComment()

Relationship with other classes:

- Resource
- Comment
- Request
- Classes which can be used during search

An end user can be of several types. Hence following are the sub classes of End User:

Student

Attributes:

- Interest

Methods:

- findUser_sameInterest()
- getInterest()
- setInterest()

Professor

Attributes:

- University
- Area of specialization

Methods:

- getUniversity()
- setUniversity()
- getArea_of_specialization
- setArea_of_specialization

Others

Conference

Attributes:

- Organization
- Venue
- Date
- ConfID
- ConfSubject
- ConfOverview

Methods:

- getOrganization()
- setorganization()
- getVenue()
- setVenue()
- getDate()
- setDate()
- getConfID()
- getConfSubject()
- setCOnSubject()
- getOverview()
- setOverview()
- getOtherPapers()

Relationship with other classes:

- Organization
- Subject
- Conference Paper

Organization

Attributes:

- Head Office
- Starting Date

Methods:

- getDate()
- setDate()
- getLocation()
- setLocation()

Subject

Attributes:

- Sub_ID
- Sub_Info
- Sub_Name

Methods:

- getName()
- setName()
- getID()
- getInfo()
- setInfo()

Digital Library Institute

Attributes:

- Name
- ID
- Website_Link
- Info

Methods:

- getName()
- setName()
- getID()
- setID()
- getLink()
- getInfo()
- setInfo()

Comment

Attributes:

- Comment_content
- Comment_by_User
- Comment_On_Rsc
- Comment_ID

Methods:

- getUser()
- getRsc()
- getContent()

Relationship with other classes:

- End User
- Resource

Request

Attributes:

- RqstID
- ResourceRqst
- Rqst_Status
- RqstMaker

Methods:

- getID()
- getResourceName()
- setResourceName()
- getStatus()
- setStatus()
- getRequestMaker()

Relationship with other classes:

- End User
- Resource
- Librarian

The digital library class model has been developed using rational rose tool and presented in Figure 16.

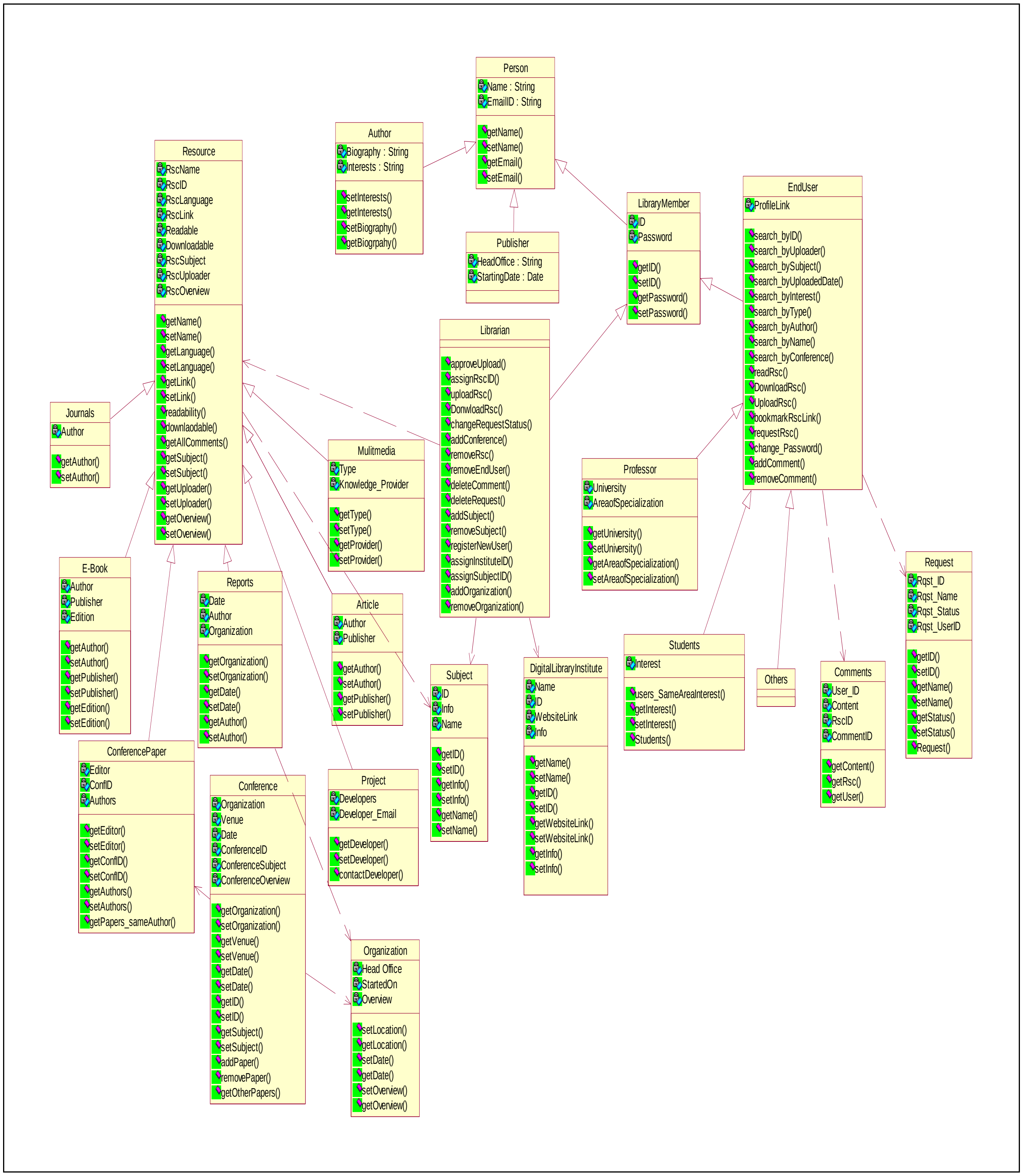


Figure 16 Class Model for Digital Library using Rational Rose Tool

4.4 Summary

The domain model developed for semantic web based digital library using UML has been presented in this chapter. Object oriented methodology has been followed to develop this domain model. Initially library member, author, librarian, publisher have been identified as different user categories. The various functionalities like uploading resources, search, and request facility for these user categories have been modeled using use case models. The UML class model for semantic web based digital library domain has been built. The various entities belonging to the problem domain have been conceptualized as classes using noun analysis technique. The methods and attributes for each of the classes and relationships among classes have been identified.

5. Domain Ontology and Agent Behaviour Modeling using UML

5.1 Domain Ontology Model using UML

The ontology contains the information about things that exists in the given problem domain. The ontology does not talk about the kind of the processing these components undergo as they take part in various functionalities. So the main difference between the UML class model and the domain ontology model is the methods or operations.

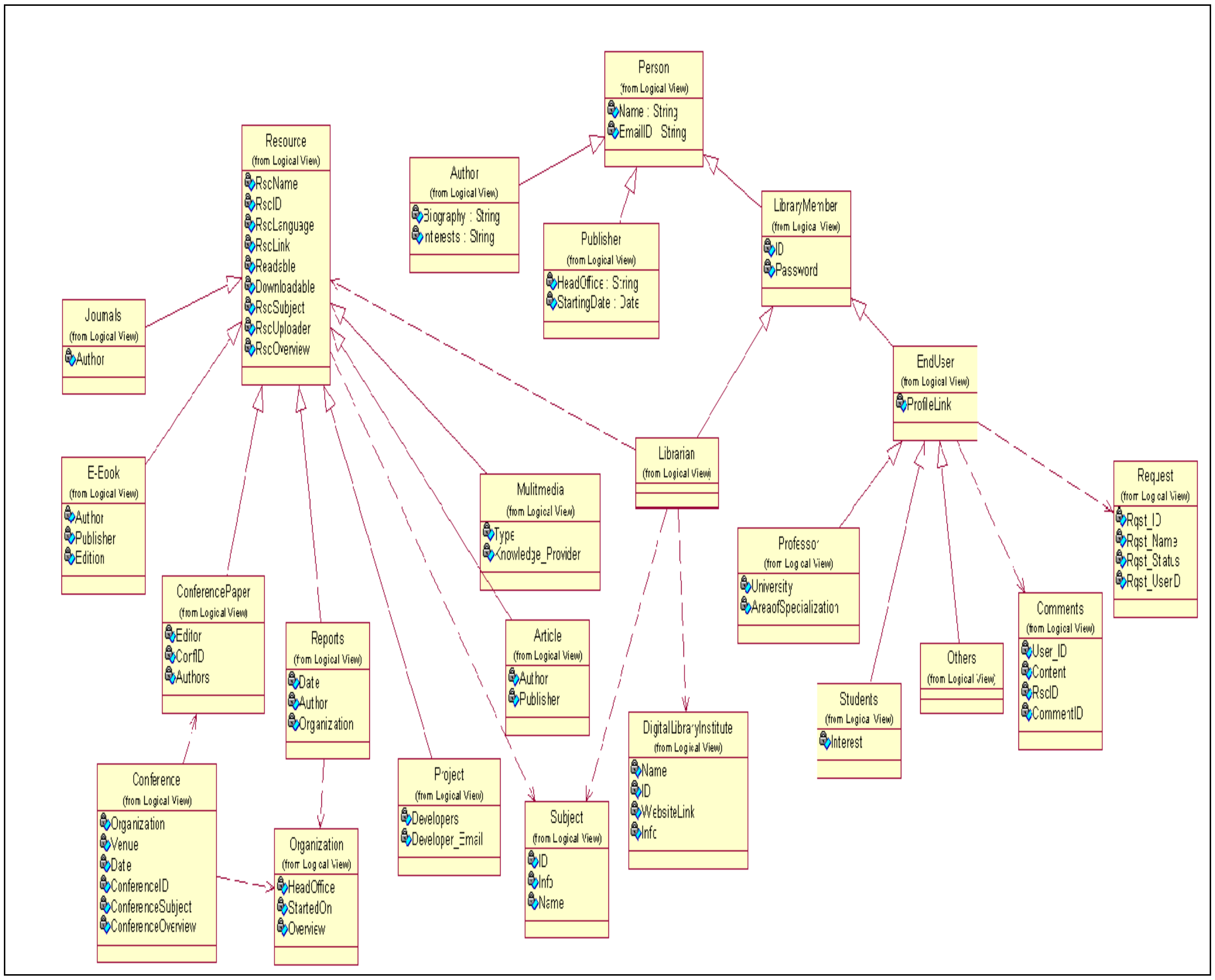


Figure 17 Domain Ontology Model of Digital Library Using Rational Rose Tool

In order to convert the UML Class Model to the corresponding Domain Ontology Model, the operations for each of the UML classes have been dropped as shown in figure 17.

5.2 OWL and UML Feature Comparison for Digital Library Domain

The Table 1 in Chapter 2 discusses the comparable features of modeling languages and OWL. This section discusses and demonstrates those similarities for the semantic web based digital library. Here for convenience, the OWL ontology has been developed using Protégé tool, it is further compared to the UML domain model.

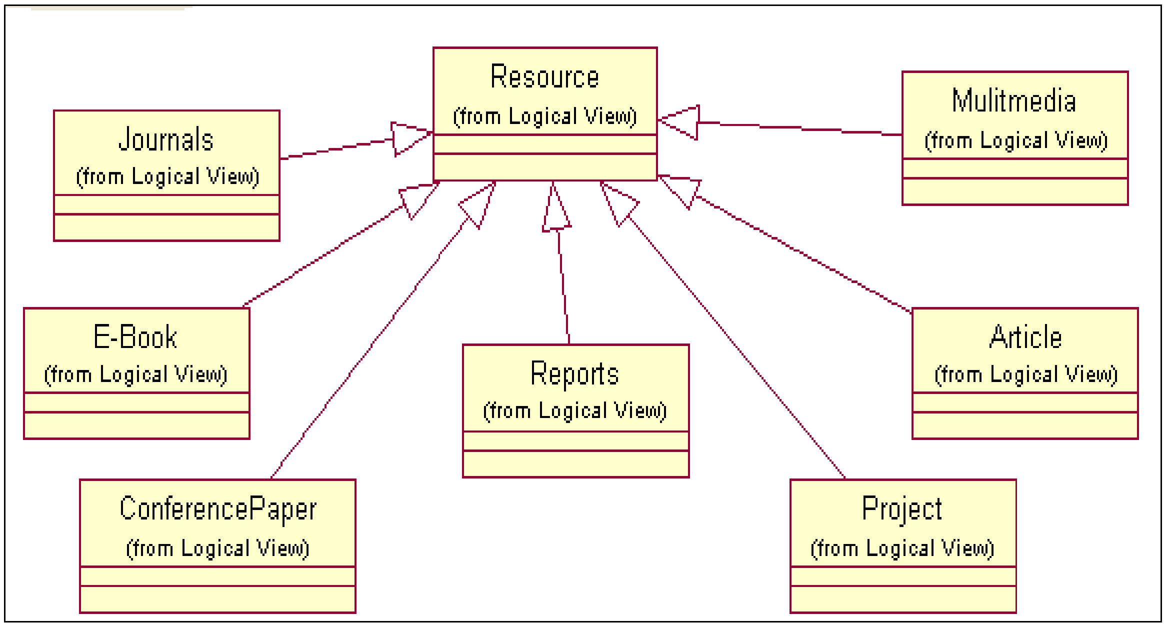


Figure 18 Part of Class Hierarchy for Resource Class using Rational Rose

5.2.1 Class (OWL/RDF) and Class (Meta-modeling Language)

Important concepts from DL domain have very similar representations in UML and OWL. While using UML as modeling language for digital library, the relevant concepts like resources are represented as UML classes which have subclasses like Journal, Multimedia, Article, Project, Conference Paper and E-book. The OWL representation of the same concept Resources has been seen as OWL class Resources with the same hierarchy of concepts following it.

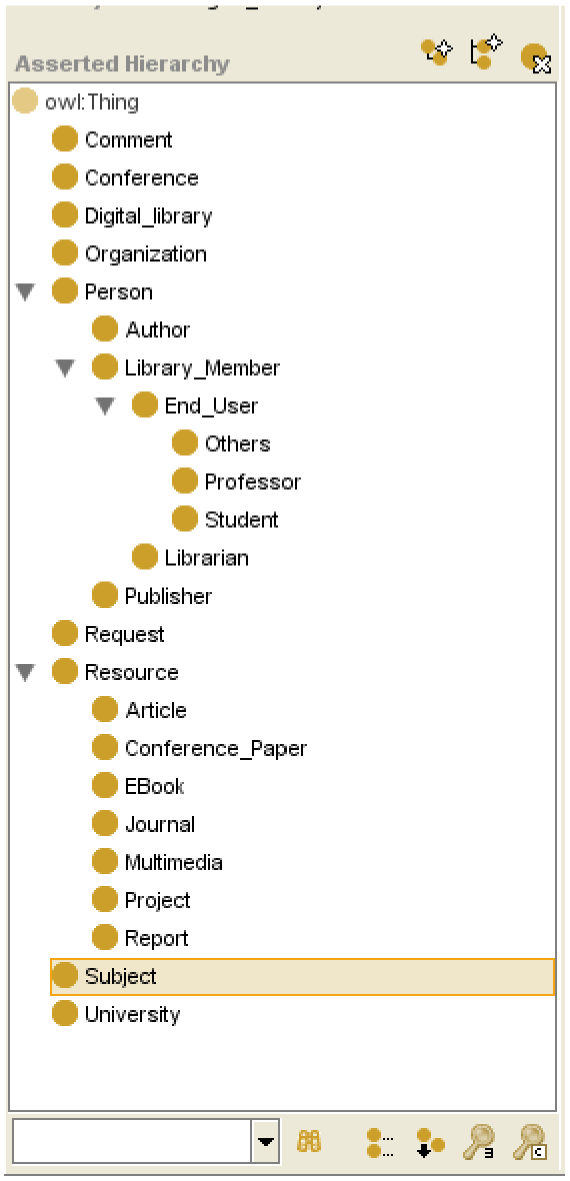


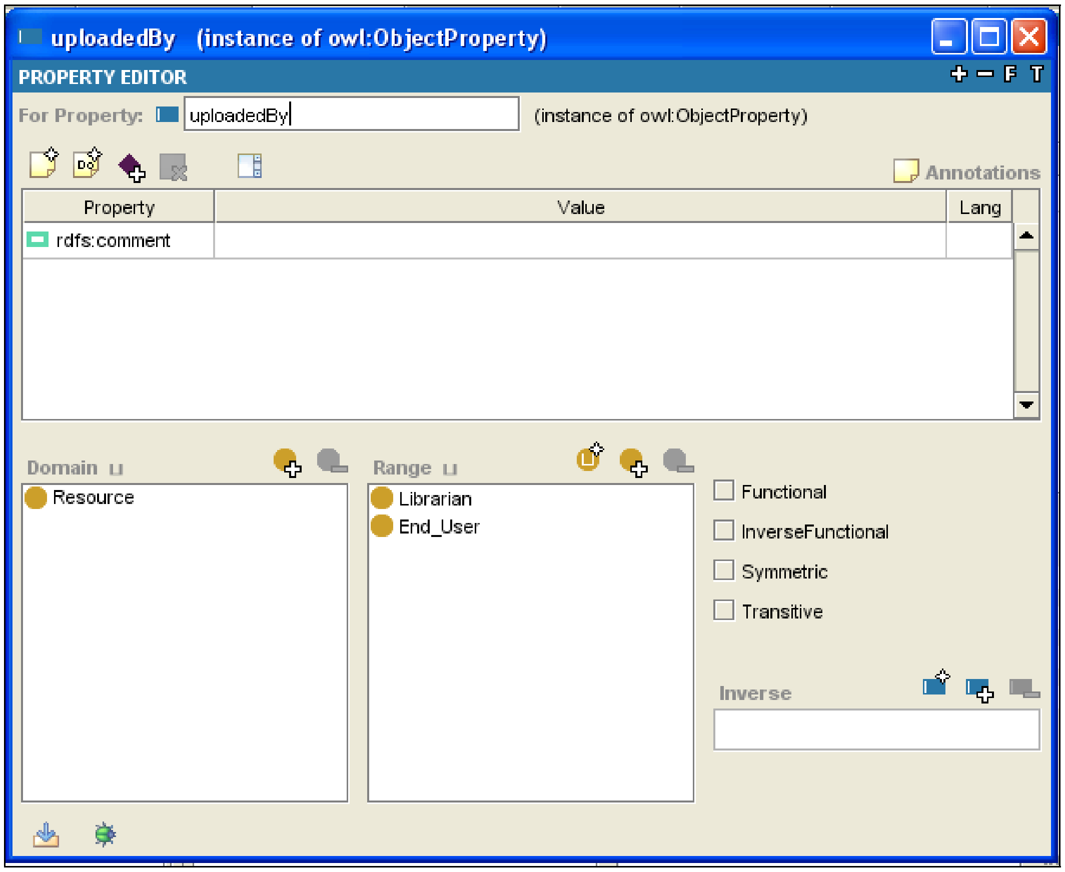
Figure 19 Class Hierarchy in Ontology using Protégé

The OWL representation of the same concept resources is seen as OWL class with the same hierarchy of concepts following it. The figures 18 and 19, show the similarity in the way concepts in the form of classes in ontology resemble the classes in the UML model.

5.2.2 Properties in OWL & Relationships in UML

There are 3 types of properties supported in OWL

1) Object Type Property
2) Data Type Property
3) Annotation Property


uploadedBy (instance of owl:ObjectProperty)

PROPERTY EDITOR

For Property: uploadedBy (instance of owl:ObjectProperty)

Annotations

Property	Value	Lang
rdfs:comment		

Domain: Resource

Range: Librarian, End_User

☐ Functional
☐ InverseFunctional
☐ Symmetric
☐ Transitive

Inverse

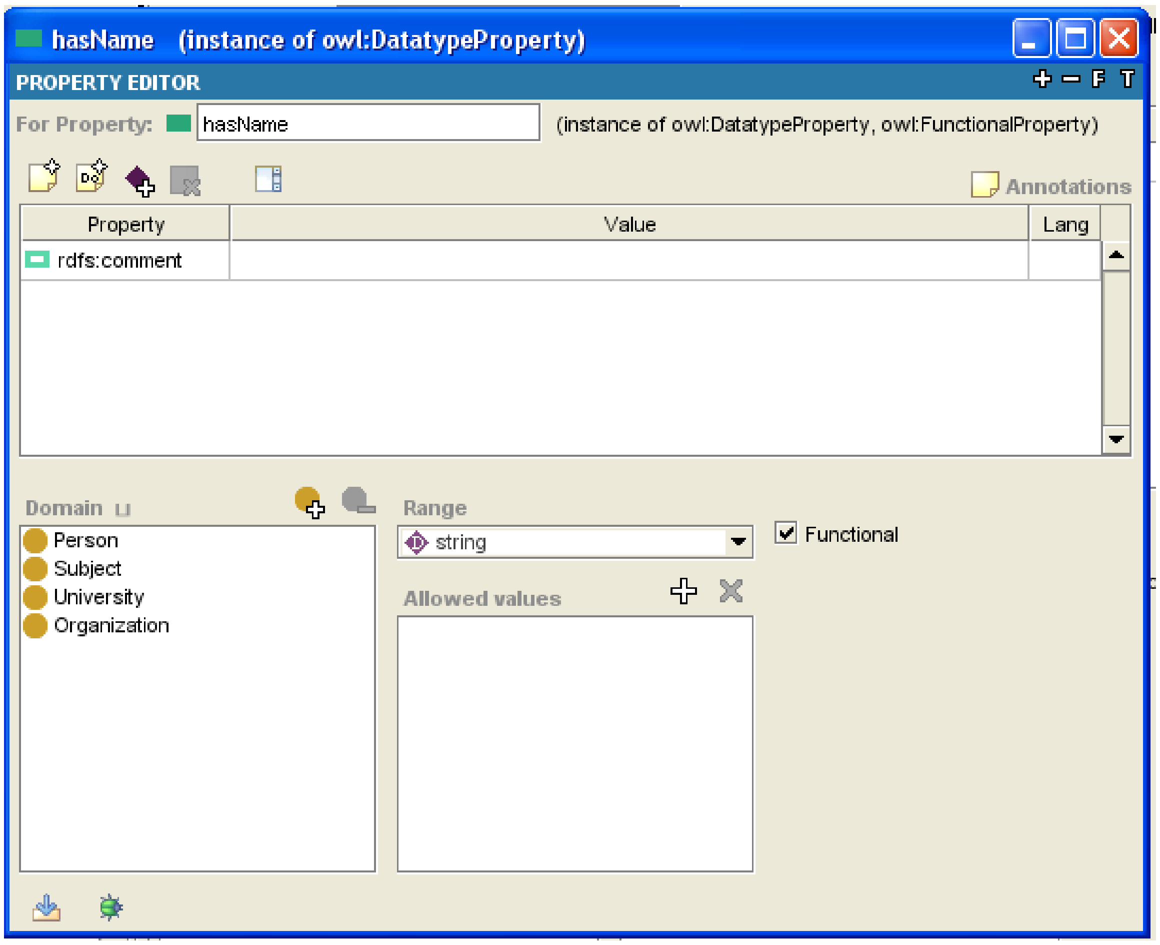
Figure 20 Object Type Property in OWL using Protégé

5.2.2.1 Object Property

This property covers the relationships between different concepts. A property is supported by domain and range. For example, in the case of digital library, consider a property: uploadedBy

"Resource is uploaded by a Library Member"

Figure 20 shows that, the domain is Resource and Range is Library Member



hasName (instance of owl:DatatypeProperty)

PROPERTY EDITOR

For Property: hasName (instance of owl:DatatypeProperty, owl:FunctionalProperty)

Property	Value	Lang
rdfs:comment		

Domain: Person, Subject, University, Organization

Range: string

Allowed values

Functional ✓

Figure 21 Data Type property in OWL with its Range and Domain

5.2.2.2 Data Type Property

The data type property has a class as its domain and its range is some data type like String, Date as seen in figure 21.

For example, consider a property hasName

“Person has a Name”.

Here domain is Person and Range is a String.

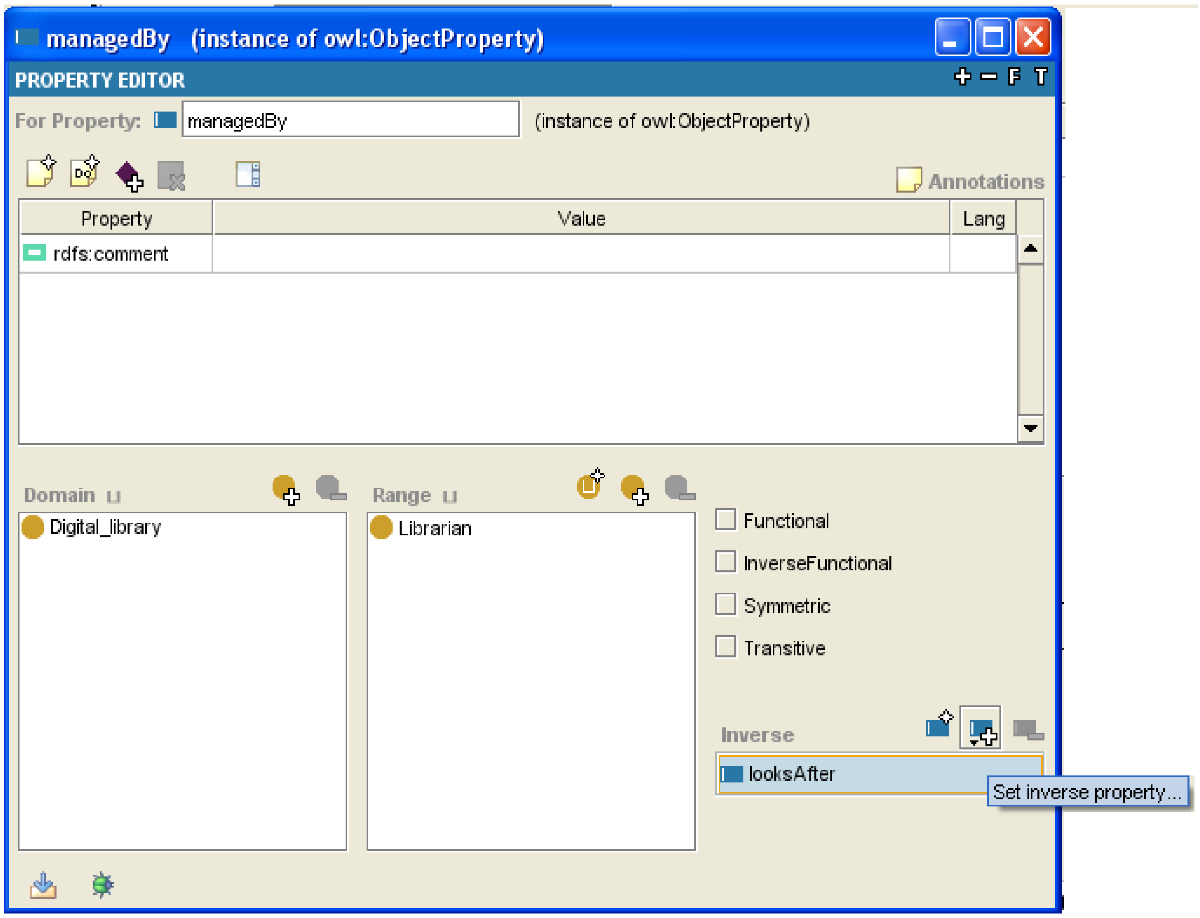


Figure 22 Showing Inverse Property in OWL using Protégé

5.2.2.3 Annotation Property

Annotation Property adds more information about a property or a class.

A property in OWL can be further defined by adding several characteristics like:

Functional: It ensures that only one individual is related with another individual. In case of data type property, a single value is taken.

For example, Organization started on a date. Here ***the property startedOn*** is set ***Functional*** as there can be only 1 unique date on which the organization started.

Transitive: A property relating A to B and then B to C is transitive if it can be inferred that A is related to C with the same property.

Inverse: A property linking individual A to B, then the inverse property will relate B to A. For Example: Digital Library is **managed by** Librarian and Librarian **looks after** Digital Library as depicted in figure 22.

5.2.3 Types of Relationships

5.2.3.1 Unidirectional Association

Consider the property: uploadedBy.

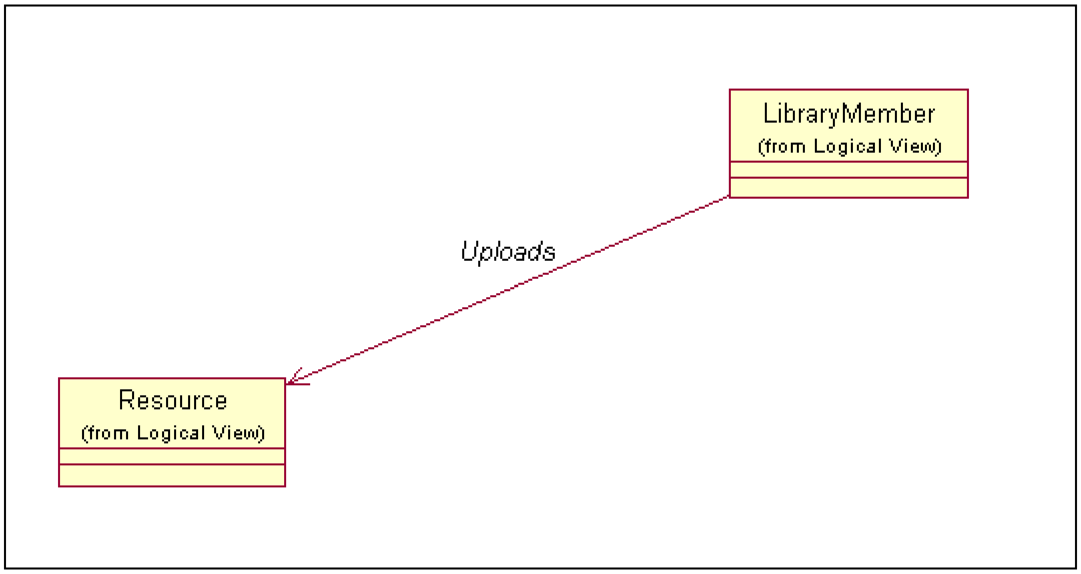


Figure 23 Unidirectional Association between Resource and Library Member

The figure 23 shows the association relationship between Resource and Library Member.

5.2.3.2 Binary Association

Consider the properties: loooksAfter and managedBy existing for classes librarian and digital library as seen in figure 24.

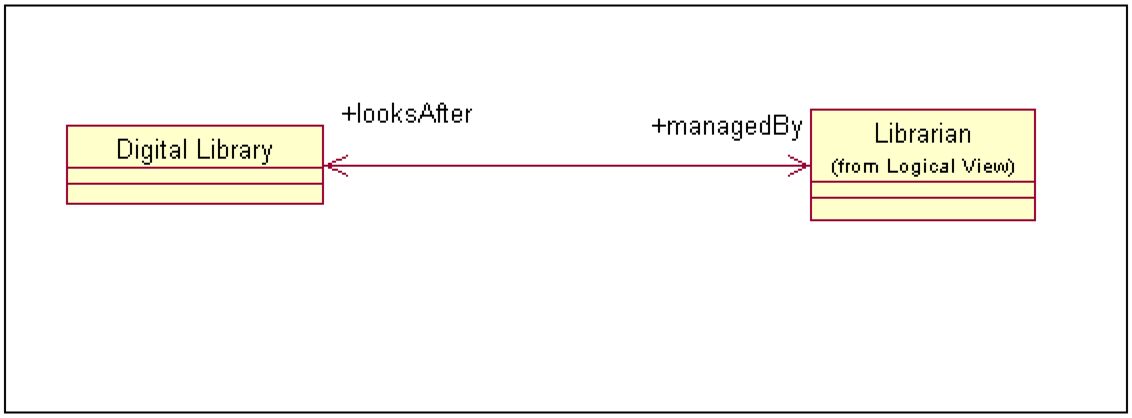


Figure 24 Bidirectional Association between Digital Library and Librarian.

5.2.3.3 Aggregation

Digital Library is part of University depicts aggregation type of relation as shown in figure 25.

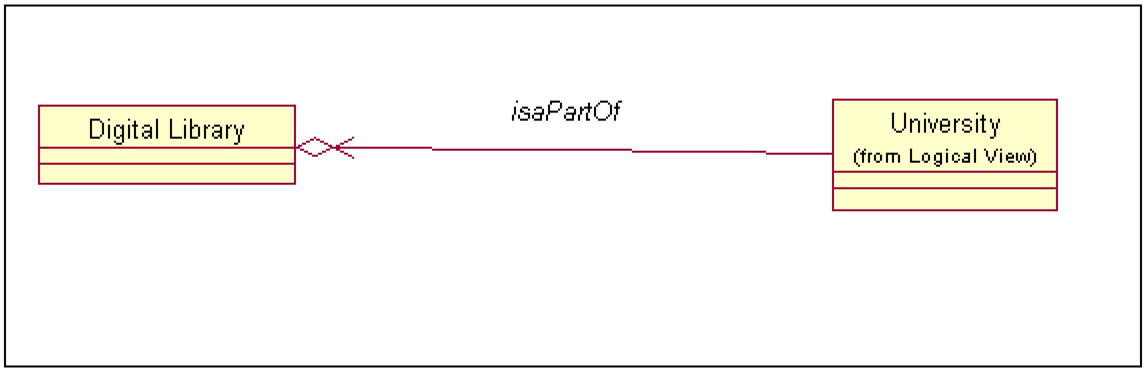


Figure 25 Aggregation Relationship between University and Digital Library

Binary Association is equivalent to an Inverse Object Type Property. A Unidirectional Association is equivalent to the Non Inverse Object Type Property.

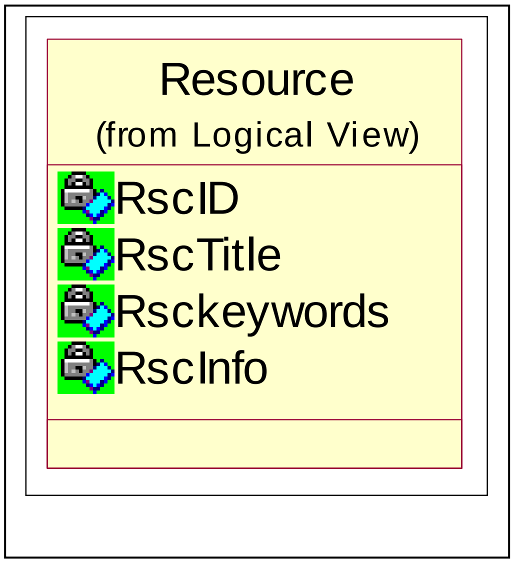


Figure 26 Attributes of Class Resource

5.2.4 Attributes in UML

Resource Class in Digital Library Problem has attributes- RscID, RscTitle, RscKeywords, RscInfo. The attributes actually resemble the Data Type Property as shown in Figure 26 and Figure 27.

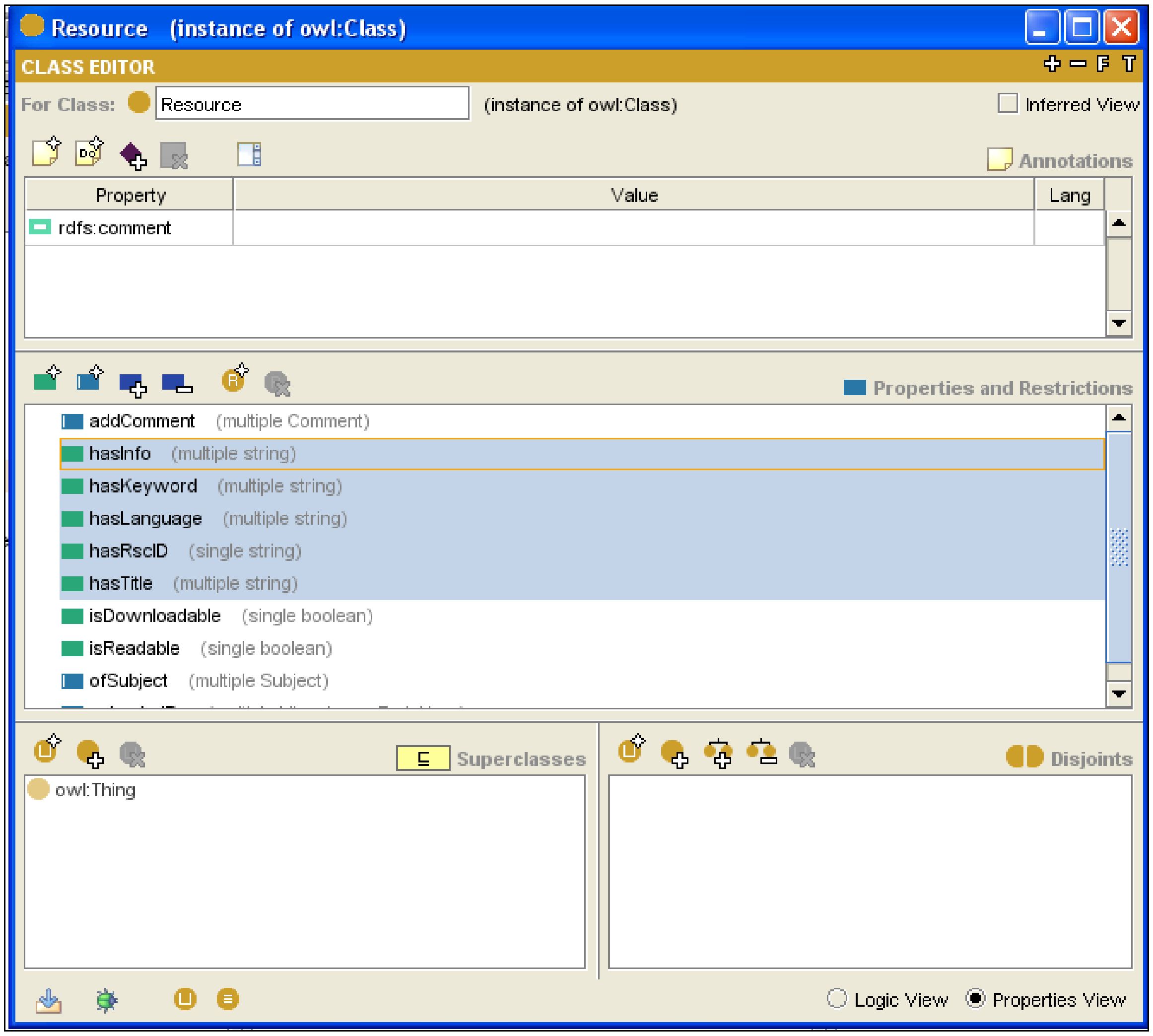
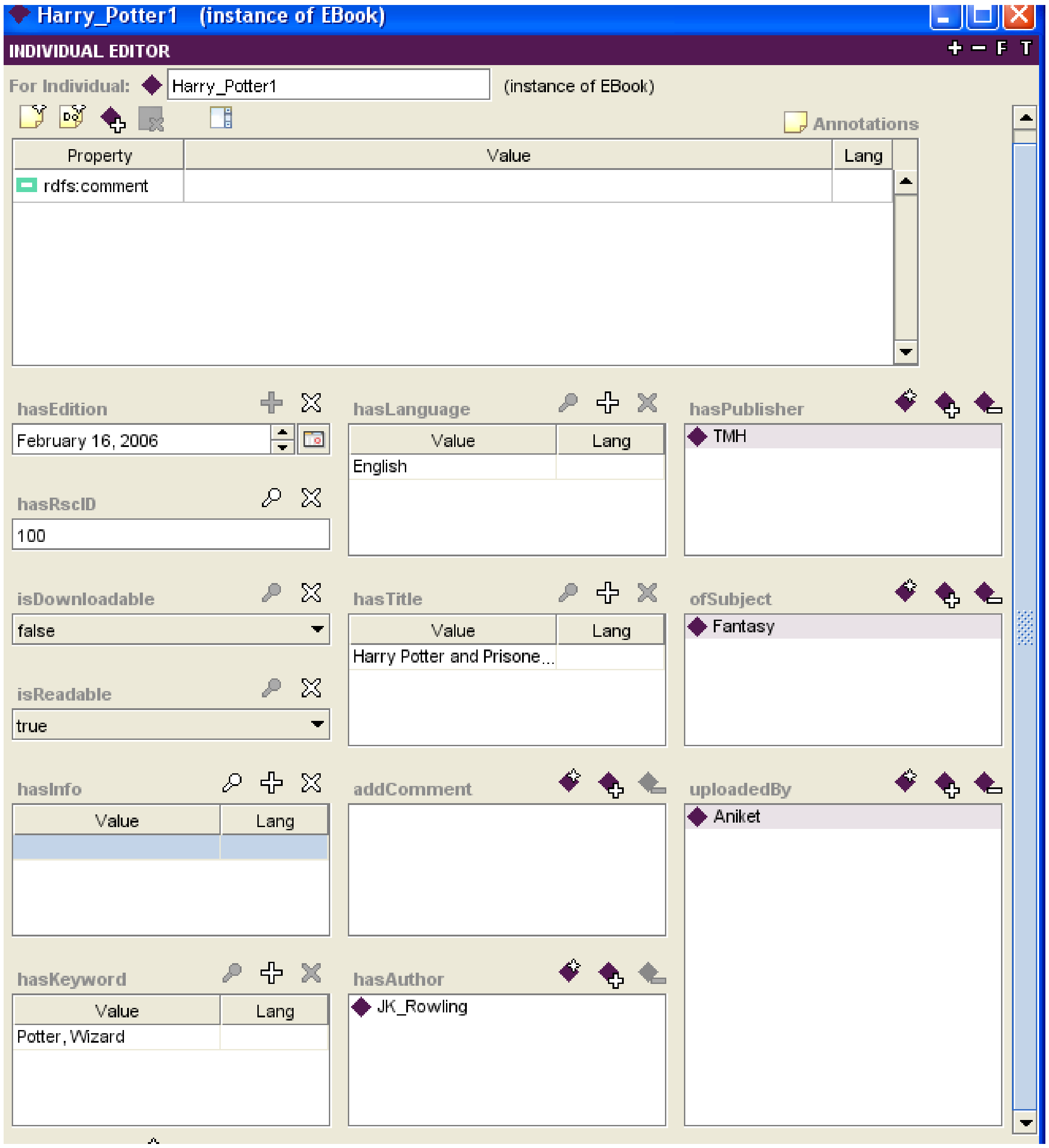

Resource (instance of owl:Class)
CLASS EDITOR
For Class:
Resource
(instance of owl:Class)
Inferred View
Annotations
Property
Value
Lang
rdfs:comment
Properties and Restrictions
addComment (multiple Comment)
hasInfo (multiple string)
hasKeyword (multiple string)
hasLanguage (multiple string)
hasRscID (single string)
hasTitle (multiple string)
isDownloadable (single boolean)
isReadable (single boolean)
ofSubject (multiple Subject)
Superclasses
owl:Thing
Disjoints
Logic View
Properties View

Figure 27 Comparison between Attributes and Data Type Properties in OWL

5.2.5 Instances in OWL



Harry_Potter1 (instance of EBook)

INDIVIDUAL EDITOR

For Individual: Harry_Potter1 (instance of EBook)

Annotations

Property	Value	Lang
rdfs:comment		

hasEdition: February 16, 2006

hasRscID: 100

isDownloadable: false

isReadable: true

hasLanguage

Value	Lang
English	

hasTitle

Value	Lang
Harry Potter and Prisone...	

hasInfo

Value	Lang

hasKeyword

Value	Lang
Potter, Wizard	

hasPublisher: TMH

ofSubject: Fantasy

addComment

uploadedBy: Aniket

hasAuthor: JK_Rowling

Figure 28 Instantiation of the Class EBook in OWL using Protégé

In a Metamodeling language like UML, once the class diagram is complete, classes are instantiated so as to represent the objects in real as shown in figure 28.

5.3 Behaviour Modeling of Digital Library Agents

Every entity that has been identified as part of the system has two aspects; structural aspect and behavioral aspect. Any system can be examined by its static structure, which is classes and relationships. The aspects of the system that are concerned with time and changes are modeled using behavioral models.

The UML use case model allows us to represent functionalities of agents in the form of use cases. This in turn will help in analyzing the agent interactions better. Semantic web agents will have to carry out the task of finding out and using the information on the web from several resources on their own, processing them and integrating the results and presenting them to the users or carrying out inferencing based on those results. The interaction model helps in analyzing the interactions among agents as they take part in query execution. The message passing among the agents can be represented as a function of time. The agents and objects taking part in interactions can also be shown. The various constraints can be represented using OCL elements. The overall functionality and flow process of the query can be analyzed by developing corresponding activity model.

This section presents Interaction Models and State Models developed for analyzing the behaviour of a semantic web based digital library using the technology described in Chapter 2.

5.3.1 Interaction Models

The sequence models are a type of interaction models. This section presents these models developed for the semantic web based digital library using Rational Rose tool.

5.3.1.1 Uploading Agent

The uploading agent makes use of end user, librarian and resource objects for uploading the resource as shown in figure 29. When the end user submits a new resource for upload, the librarian accesses the temporary file created. The librarian either accepts or rejects the proposal based on its authenticity. The end user is informed and the file is uploaded in case of acceptance.

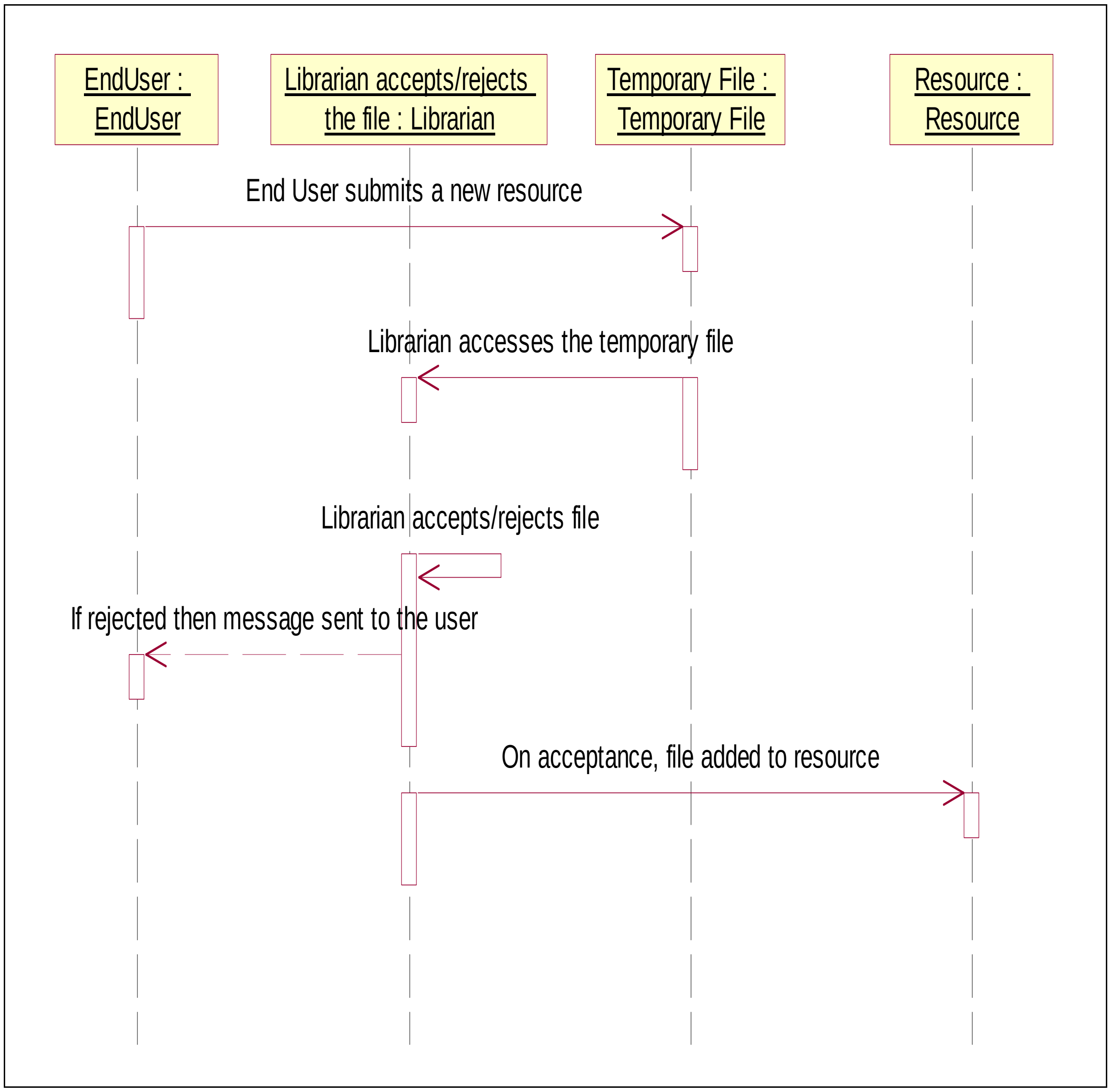

EndUser : EndUser
Librarian accepts/rejects the file : Librarian
Temporary File : Temporary File
Resource : Resource
End User submits a new resource
Librarian accesses the temporary file
Librarian accepts/rejects file
If rejected then message sent to the user
On acceptance, file added to resource

Figure 29 Upload Resource Agent Sequence Diagram using Rational Rose

5.3.1.2 Searching Agent

Searching agent helps in searching the required resources. For a standalone digital library this agent may not be very crucial but for the semantic web based federated digital library that has been modeled in this thesis, the searching over distributed digital collections is not very straight forward. The interaction model helps in analyzing the searching agent behaviour. When the end user submits search parameters, the parameters are mapped to concepts as in digital library ontology by search agent. The resource database is searched for the required matching concepts using concept relationship mappings based on spread activation techniques and matches are mapped to the actual documents which are then presented to user as a result set as shown in figure 30.

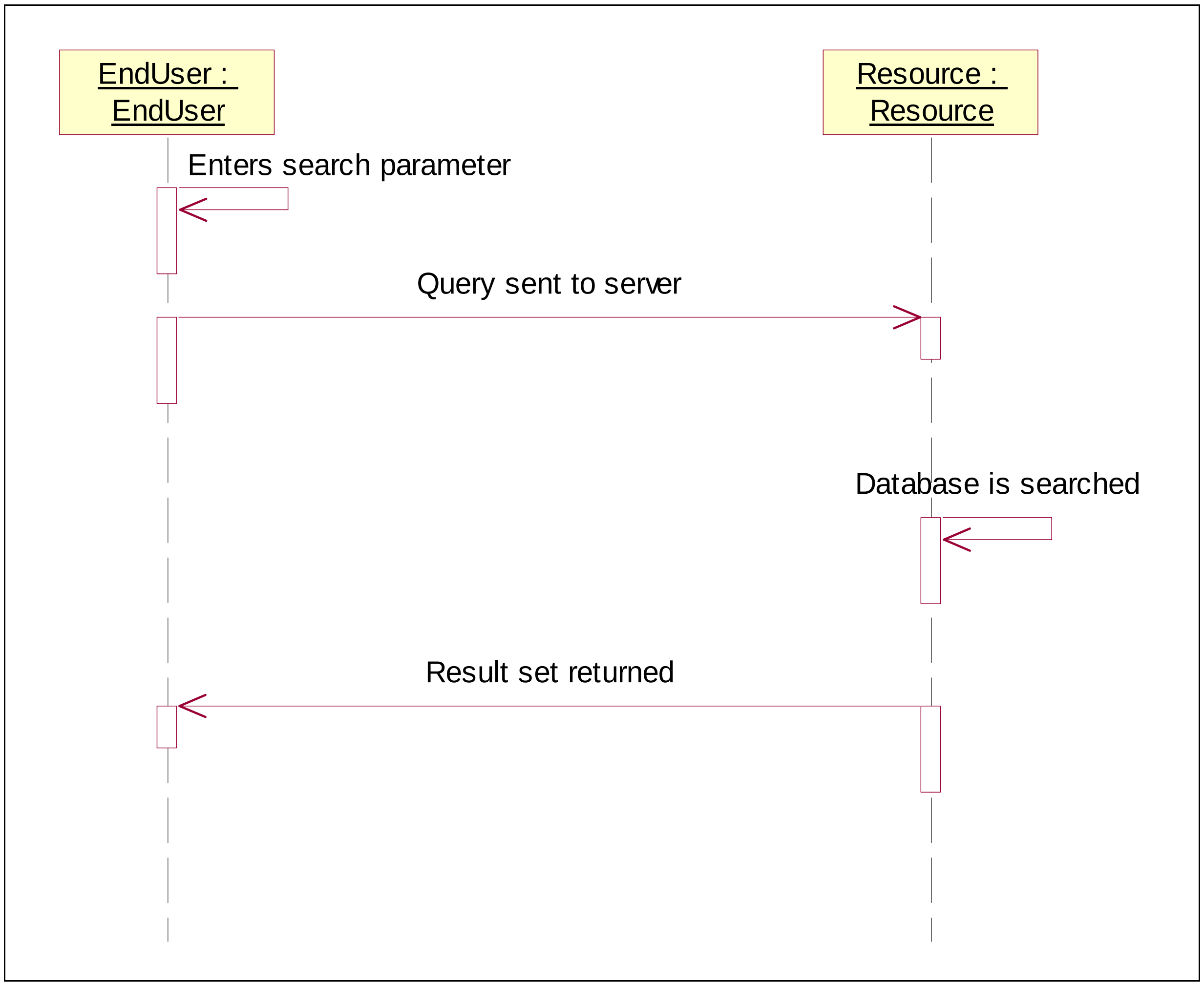


Figure 30 Searching Agent Sequence Diagram Using Rational Rose

5.3.2 State Models

The agents change states in response to the events or processing they undergo. The objects of the same class may respond differently to the same set of events.

5.3.2.1 Request State Model

A specific request for a resource can be sent. As in figure 31. The librarian accepts or rejects depending on the authenticity of the user and the availability of the resource.

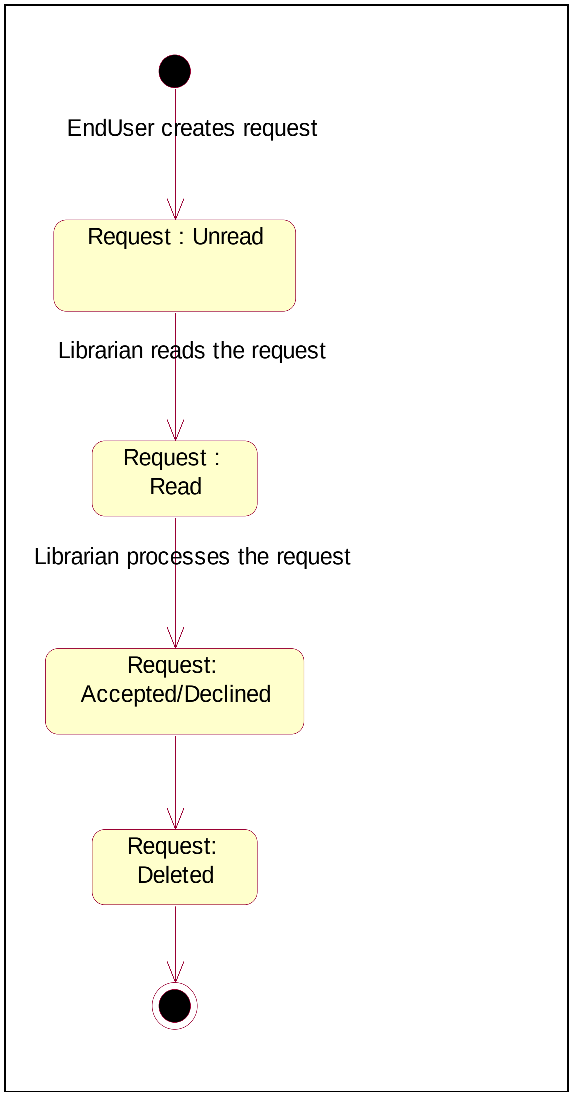

EndUser creates request
Request : Unread
Librarian reads the request
Request : Read
Librarian processes the request
Request: Accepted/Declined
Request: Deleted

Figure 31 Request Agent State Model using Rational Rose

5.3.2.2 Resource State Model

The state diagram shown for resource class depicts the state changes that it goes through when it is used by upload agent. The temporary resource file created by the end user will be made permanent by the librarian after doing authentication checks for the content and the end user as shown in figure 32. This resource will be uploaded if found acceptable and will be made available to the users of the digital library for searching and browsing.

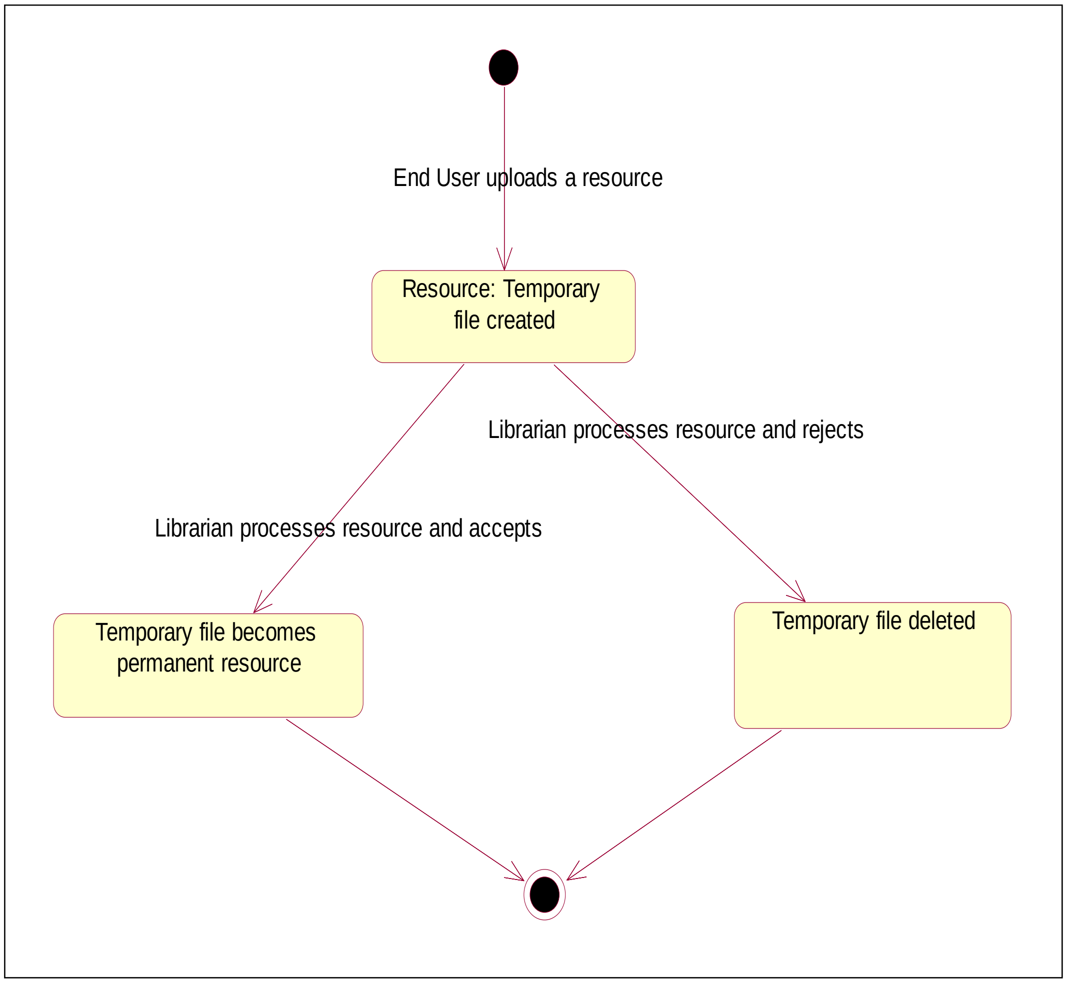

End User uploads a resource
Resource: Temporary file created
Librarian processes resource and rejects
Librarian processes resource and accepts
Temporary file becomes permanent resource
Temporary file deleted

Figure 32 Resource Agent State Model using Rational Rose

5.4 Summary

The domain ontology model for the digital library is obtained by dropping methods from the UML class model. Various UML modeling constructs appearing in class model like classes, relationships are mapped to corresponding modeling constructs of OWL. Rational Rose tool and Protégé are used to demonstrate these modeling construct mappings which are based on transformations defined in Table 1. UML behavioural models are used to model behavioural aspects of semantic web digital library agents. The interaction models are presented for searching and uploading agents in this chapter. The state models represent state changes an object undergoes in response to events. The state changes for request and resource agent are presented in this chapter.

6. Automation of Semantic Web Based Digital Library

The process of building software systems using object oriented paradigm which uses the software engineering models was benefited because of the analysis and design capabilities of these models. But as more and more such systems are built and the vision of semantic web based systems comes into reality, the whole process of ontology building needs to be automated. Although visual models are very helpful when it comes to building and maintaining ontologies, implementation of these models in the OWL is not an easy job. The maintenance of the implemented system is even a tougher job. If the ontology generation can be made completely automated it will be a major achievement for development of semantic web based systems. We have used MOF based UML and MDA to build a digital library system. The use of MOF based visual models make the updation of ontologies very easy. The research methodology discussed in Chapter 3 can generate Java and OWL implementations of these MOF based models [Bhi2009b]. The XSLT mapping that has been used generates implementation of the MOF based models in Java and OWL. These OWL ontologies are in a format that is understandable by intelligent agent which will actually do all the data collection and inferencing necessary to answer a semantic query posed by semantic web user.

6.1 Our Approach

The approach followed in the presented work has been shown in figure 33. The XSLT mapping created for the UML model can generate the OWL code. The UML class model for digital library domain has been created using the UML Rational Rose tool. . From this .mdl format model the XMI coding has been generated by the rational rose tool itself. The XSLT transformations for OWL have been developed using the mappings defined by Djuric [DGD2005]. The rational rose tool has inbuilt XSLT mapping of Java code available.

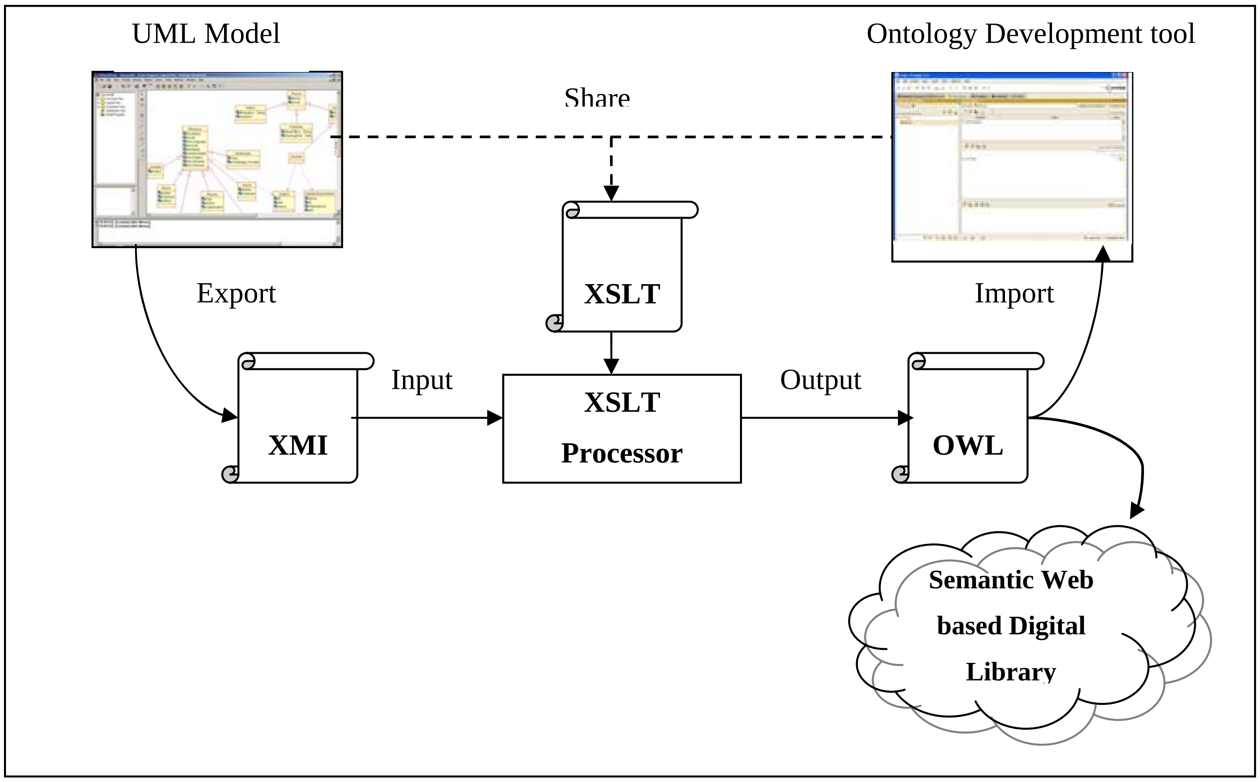


Figure 33 OWL Mapping for Semantic Web based Digital Library

6.2 Java Code Generation from Digital Library Ontology Model

The Rational Rose tool generates Java templates for the input UML class model. The method for which has been discussed in Chapter 3. The Author hierarchy for which the Java code has been generated using rational rose tool is shown in figure 34. Here Author is a subclass of superclass Person. The snapshot of the fragment from the generated code has been shown in figure 35 for this hierarchy [Bhi2009b]. The complete Java code generated for the domain model of figure 17 has been documented in appendix A.

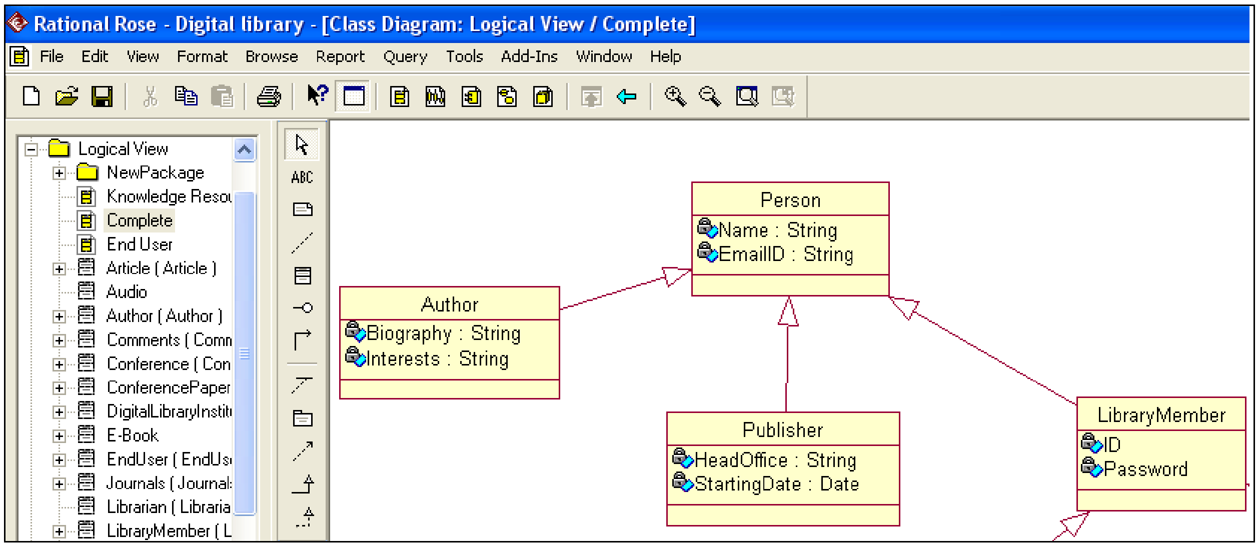
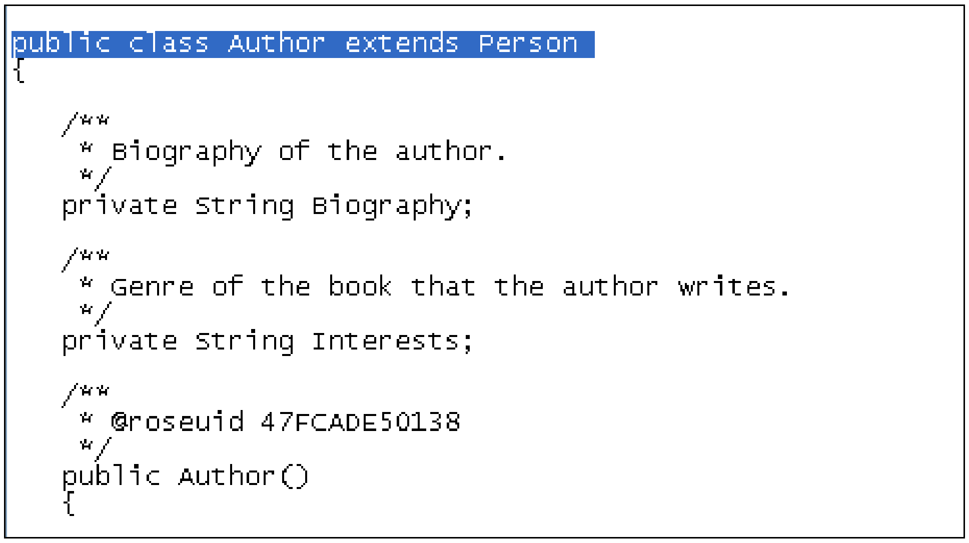


Figure 34 Author Hierarchy from Class Model Using Rational Rose Tool



```
public class Author extends Person
{

    /**
     * Biography of the author.
     */
    private String Biography;

    /**
     * Genre of the book that the author writes.
     */
    private String Interests;

    /**
     * @roseuid 47FCADE50138
     */
    public Author()
    {
```

Figure 35 Java Code Fragment for Author Hierarchy [Bhi2009b]

6.3 OWL Code Generation from Digital Library Ontology Model

Using the mappings described in Table 1 of Chapter 2**,** the ontology model of figure 17 has been mapped to the OWL code. The transformation of model to the code in automated fashion is very important for the future semantic web based applications. The ability to generate the code automatically will enable the semantic web digital library to work in automated way. Any changes in the domain ontology will need to make those changes in the UML domain model using a tool like rational rose. The corresponding changes will be reflected in the OWL coding that is visible to agents in automated way. Any changes in the implementation technology only demand to rewrite the corresponding XSLT mapping. The rest of the modules shown in figure 36 remain unchanged. The generated OWL code can be used by future semantic web agents for inferencing and making decisions directly. The Java code can be used by Java applications directly. The participation of domain experts for any semantic web based system will greatly enhance the quality of the generated ontologies.

The rational rose tool has been used to develop the domain ontology model for the semantic web based digital library. The generated .mdl model can be used to create the XMI format model by the rational rose tool itself. XALAN2.7 XSLT processor has been used to produce the XSLT mapping based on Table 1, Chapter 2. Using the XMI code and XSLT for OWL, OWL code for the model has been generated [Bhi2008]. Figure 37 shows the rational rose generated XMI coding of Author hierarchy shown in figure 34. The XSLT OWL transformations shown in figure 38 produce OWL code of figure 39.

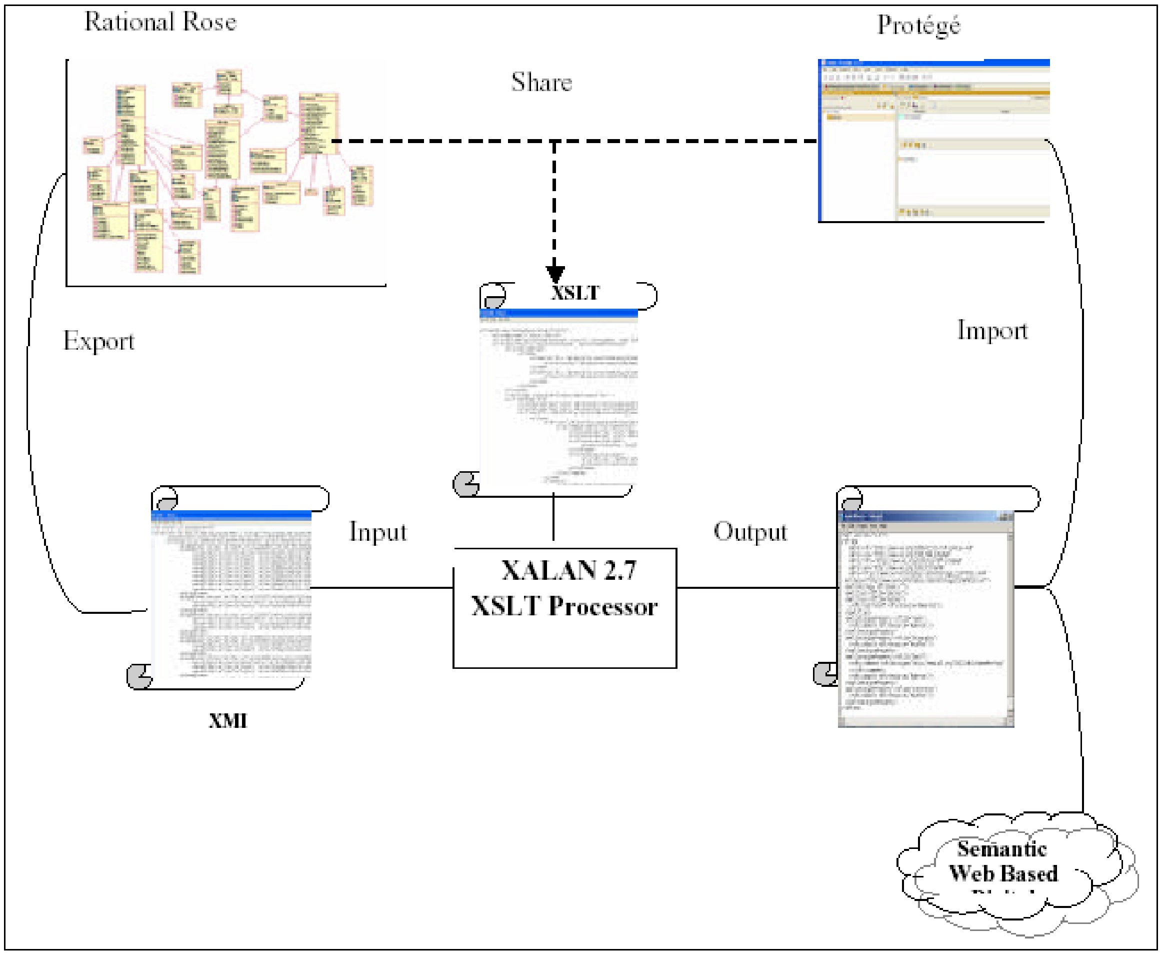
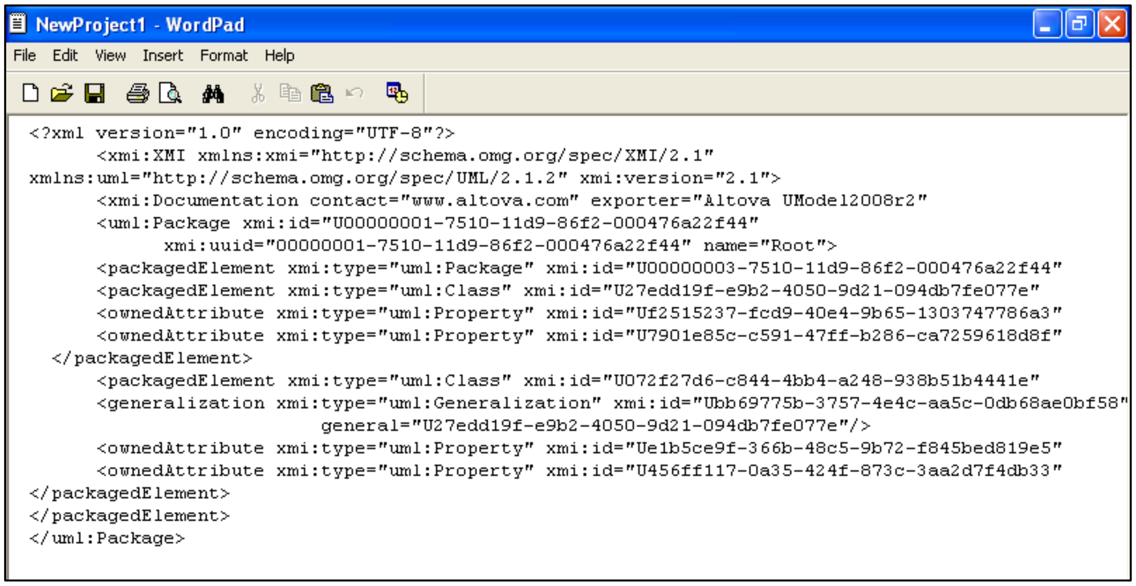
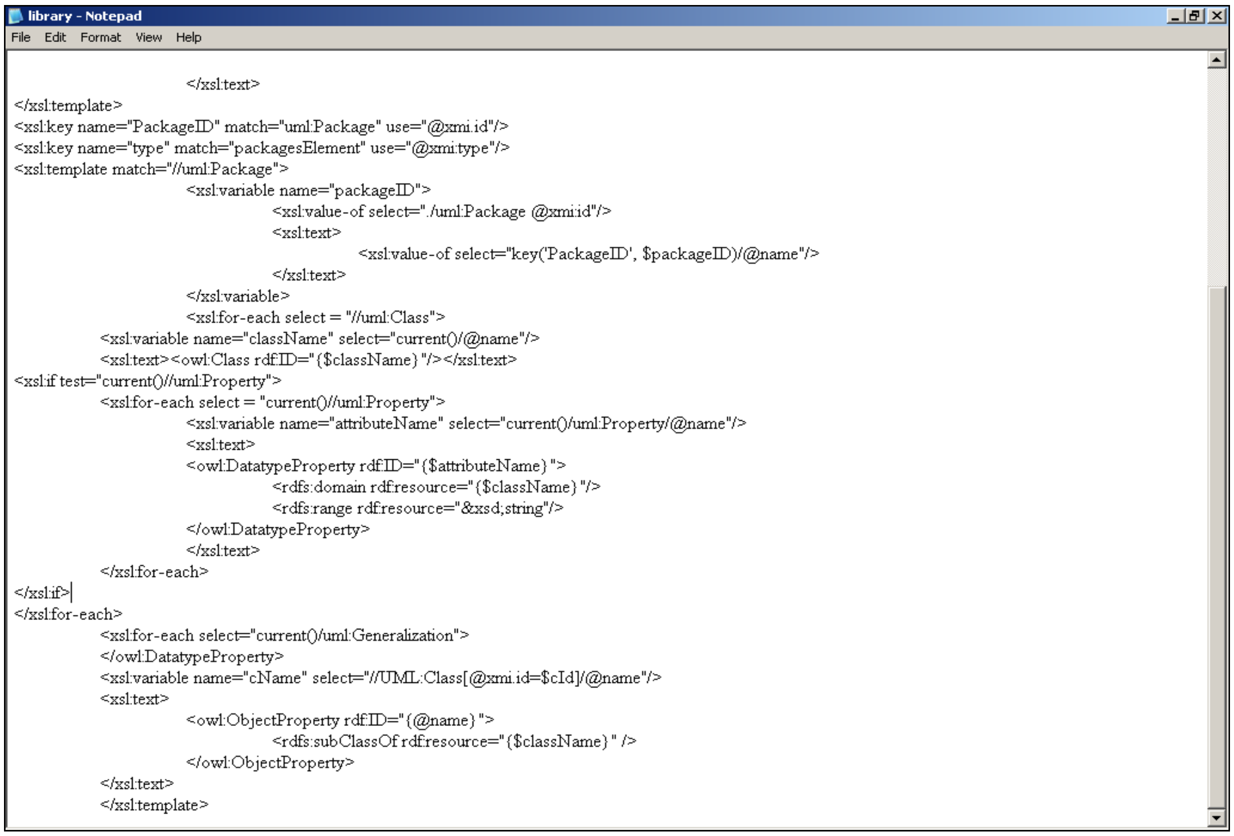


Figure 36 Technology for UML to OWL Mapping for Digital Library



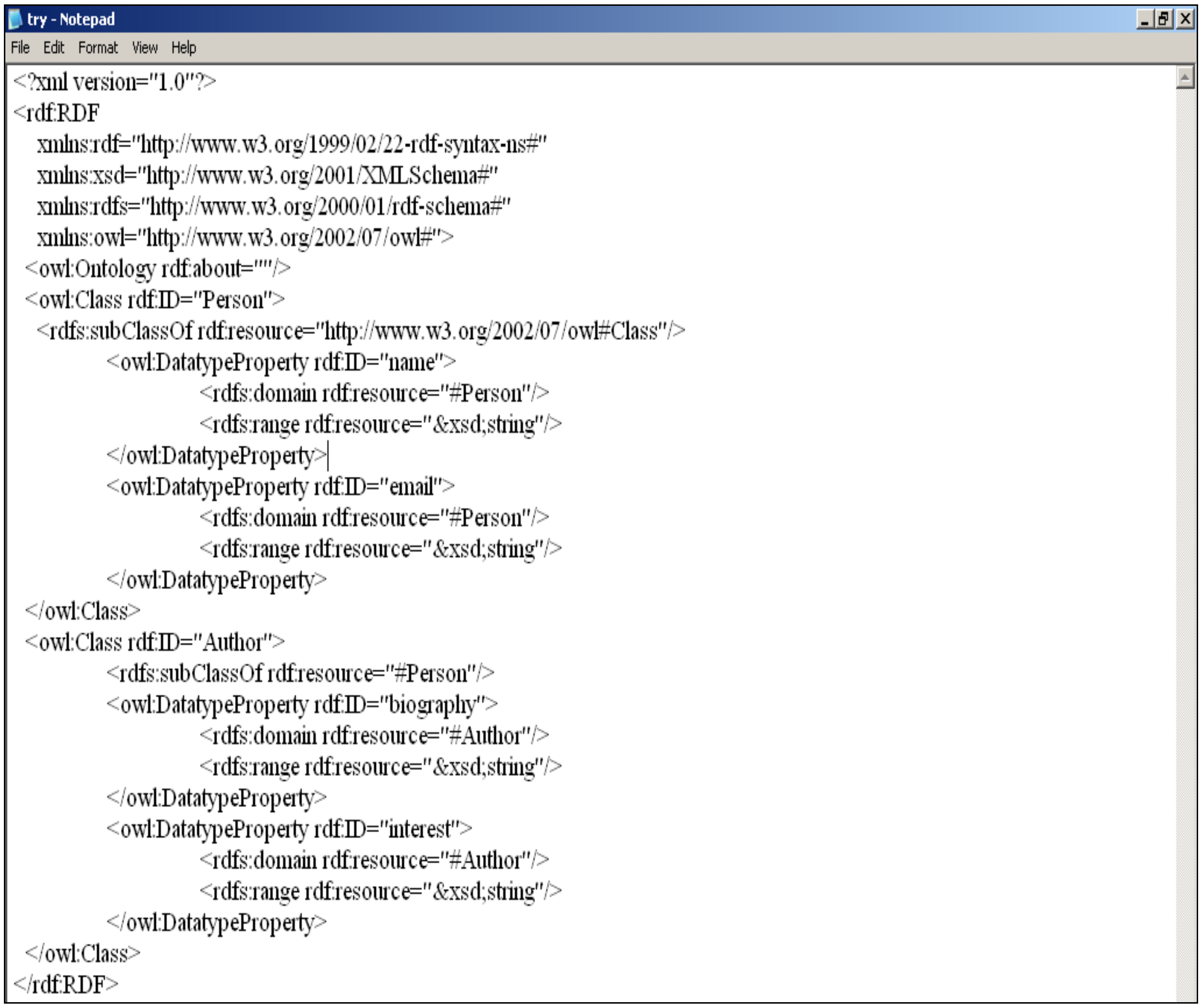
```
<?xml version="1.0" encoding="UTF-8"?>
      <xmi:XMI xmlns:xmi="http://schema.omg.org/spec/XMI/2.1"
xmlns:uml="http://schema.omg.org/spec/UML/2.1.2" xmi:version="2.1">
      <xmi:Documentation contact="www.altova.com" exporter="Altova UModel2008r2"
      <uml:Package xmi:id="U00000001-7510-11d9-86f2-000476a22f44"
            xmi:uuid="00000001-7510-11d9-86f2-000476a22f44" name="Root">
      <packagedElement xmi:type="uml:Package" xmi:id="U00000003-7510-11d9-86f2-000476a22f44"
      <packagedElement xmi:type="uml:Class" xmi:id="U27edd19f-e9b2-4050-9d21-094db7fe077e"
      <ownedAttribute xmi:type="uml:Property" xmi:id="Uf2515237-fcd9-40e4-9b65-1303747786a3"
      <ownedAttribute xmi:type="uml:Property" xmi:id="U7901e85c-c591-47ff-b286-ca7259618d8f"
  </packagedElement>
      <packagedElement xmi:type="uml:Class" xmi:id="U072f27d6-c844-4bb4-a248-938b51b4441e"
      <generalization xmi:type="uml:Generalization" xmi:id="Ubb69775b-3757-4e4c-aa5c-0db68ae0bf58"
                    general="U27edd19f-e9b2-4050-9d21-094db7fe077e"/>
      <ownedAttribute xmi:type="uml:Property" xmi:id="Ue1b5ce9f-366b-48c5-9b72-f845bed819e5"
      <ownedAttribute xmi:type="uml:Property" xmi:id="U456ff117-0a35-424f-873c-3aa2d7f4db33"
</packagedElement>
</packagedElement>
</uml:Package>
```

Figure 37 Snapshot of XMI for Author hierarchy using Rational Rose



```
                        </xsl:text>
</xsl:template>
<xsl:key name="PackageID" match="uml:Package" use="@xmi.id"/>
<xsl:key name="type" match="packagesElement" use="@xmi:type"/>
<xsl:template match="//uml:Package">
                        <xsl:variable name="packageID">
                                    <xsl:value-of select="./uml:Package @xmi:id"/>
                                    <xsl:text>
                                                <xsl:value-of select="key('PackageID', $packageID)/@name"/>
                                    </xsl:text>
                        </xsl:variable>
                        <xsl:for-each select = "//uml:Class">
            <xsl:variable name="className" select="current()/@name"/>
            <xsl:text><owl:Class rdf:ID="{$className}"/></xsl:text>
<xsl:if test="current()//uml:Property">
            <xsl:for-each select = "current()//uml:Property">
                        <xsl:variable name="attributeName" select="current()/uml:Property/@name"/>
                        <xsl:text>
                        <owl:DatatypeProperty rdf:ID="{$attributeName}">
                                    <rdfs:domain rdf:resource="{$className}"/>
                                    <rdfs:range rdf:resource="&xsd;string"/>
                        </owl:DatatypeProperty>
                        </xsl:text>
            </xsl:for-each>
</xsl:if>
</xsl:for-each>
            <xsl:for-each select="current()/uml:Generalization">
            </owl:DatatypeProperty>
            <xsl:variable name="cName" select="//UML:Class[@xmi.id=$cId]/@name"/>
            <xsl:text>
                        <owl:ObjectProperty rdf:ID="{@name}">
                                    <rdfs:subClassOf rdf:resource="{$className}" />
                        </owl:ObjectProperty>
            </xsl:text>
            </xsl:template>
```

Figure 38 Snapshot of XSLT for OWL for Author Hierarchy [Bhi2009a]



```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:owl="http://www.w3.org/2002/07/owl#">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="Person">
   <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
          <owl:DatatypeProperty rdf:ID="name">
                    <rdfs:domain rdf:resource="#Person"/>
                    <rdfs:range rdf:resource="&xsd;string"/>
          </owl:DatatypeProperty>
          <owl:DatatypeProperty rdf:ID="email">
                    <rdfs:domain rdf:resource="#Person"/>
                    <rdfs:range rdf:resource="&xsd;string"/>
          </owl:DatatypeProperty>
  </owl:Class>
  <owl:Class rdf:ID="Author">
          <rdfs:subClassOf rdf:resource="#Person"/>
          <owl:DatatypeProperty rdf:ID="biography">
                    <rdfs:domain rdf:resource="#Author"/>
                    <rdfs:range rdf:resource="&xsd;string"/>
          </owl:DatatypeProperty>
          <owl:DatatypeProperty rdf:ID="interest">
                    <rdfs:domain rdf:resource="#Author"/>
                    <rdfs:range rdf:resource="&xsd;string"/>
          </owl:DatatypeProperty>
  </owl:Class>
</rdf:RDF>
```

Figure 39 OWL Code for Author Hierarchy using XSLT Mapping [Bhi2009a]

6.4 Summary

In this chapter describes transformation of UML models to the corresponding implementations in Java and OWL codes. The UML models have XMI representations. Using XSLT mappings these XMI representations can be converted to Java and OWL codes. The chapter demonstrates conversion of ontology model for semantic web based digital model to the OWL ontologies which can be understood and used by semantic web agents. The UML models which are mapped to Java representations can be used by applications directly. XALAN 2.7 XSLT processor is used for the mappings. XSLT mappings are based on Djuric’s work [DGD2005].

7. Discussion and Future Work

7.1 Semantic Web Ontology Modeling

Semantic web ontologies have been developed in section 2, chapter 3. The ontology developing tools available in the market lack visual interfaces. The UML tool used for the presented work made the ontology generation easy because of the visual interface. The iterative object oriented methodology followed for development of digital library ontology development made the development process easy. The biggest obstacle in the way to map UML to ORL like OWL is the notion of Property. The notion of Property in KR languages corresponds to the notion of Association in UML. A Property is a first class element in all ORLs. This means that a Property can exist as an independent element of a language without being attached to Classes. But the UML Associations are connected to classes with association ends. So Association does not have independent existence. It also means that each association is unique. Thus, UML Associations express local restrictions while ORL Property can represent global restrictions. The correction to this problem has been suggested by Baclawaski [B+2001a] in the form of MOF specification which has notions of Property and Restrictions. Both Property and Restriction are modeled as UML Classifiers, enabling them to be first-class modeling primitives in UML. Also a Property has an aggregation of zero or more Association Ends, so it can exist without Class. The mapping for ORL that have been used in this thesis [DGD2007] have been based on Baclawaski's work [B+2001b].

Expressing constraints is a major problem while using UML models. This problem is partially solved by using OCL. For the future semantic web based systems OCL may not be sufficient to express complex constraints which will be further used by intelligent agents for inferencing while processing complex queries over distributed resources. The other option available is to make use of description logics. Use of description logics has inherent limitation when it comes to ease of use by domain experts [SHB2006].

7.2 Semantic Web Agent Behaviour Modeling

The agents of semantic web based system offer scalability required for web based systems. The basic functionalities of these agents have been analyzed using various behavioural UML models like sequence and state models in chapter 5. The basic functionalities expected from agents have been presented in section 2, chapter 4 in the form of use case diagrams. The interaction models have been developed for the uploading and searching agents. To increase the completeness of the generated ontologies one can make use of the behavioral UML models like interaction and activity diagrams. As an iterative object oriented modeling process has been followed, the initial class model has been refined into model in figure 16 using interaction diagrams for searching and uploading agents. The searching and uploading agent sequence models can give information about the responsibilities and roles of these agents and the classes they invoke from the ontologies. Use of other UML behaviour models like activity model has not yet been made for SW based system modeling. The activity model can be used to model overall activities of an agent.

7.3 MDA as Modeling Language for Semantic Web

The MDA architecture of OMG has been used to model the semantic web based digital library. It is a four-layer architecture as shown in figure 5. The implementation till layer M1 has been presented in chapter 3, 4, and 5. The use of MDA architecture allows to separate implementation details from the business model. For example: if we need to implement the same digital library domain model of figure 16 in another implementation language, only the code generated at M0 layer needs to be changed. MDA will drive the generated ontologies to become more formal and hence support the agent inferencing better. The XMI standard of OMG can serialize the MOF based metamodels like UML Profile and models like UML into plain XML text which is readable by any platform specific implementation. This makes exchange of metamodels and models possible. Thus it is not necessary to repeat the process of modeling an application or system functionality and behaviour each time a new technology comes along. MDA is driving UML more and more formal, so it helps in expressing complex constraints and automatic inferencing by agents. But still no commercial MDA tools are available which can process the models at M3 and M2 layers. The rational rose UML tool that has been used supports the MDA models till M1 layer very well [DGD2005, Bhi2008]. In the future, the present work can be extended by developing a complete MDA based model for the semantic based digital library. This will completely separate the business model from the implementation details.

The MDA based models like UML can be queried directly which will help us to pose queries at the higher level of abstraction. Using the OMG's upcoming Query View Transformation (QVT), the system will be able to process the queries based on MDA models. This will allow the users to query at higher abstraction level than possible at present.

7.4 W3C and OMG Approaches

The W3C has recently proposed use of semantic web technologies in the form of *Ontology Driven Architecture* for the development of object-oriented systems [PU2006, SHB2006]. The semantic web technologies like Description Logic DL and OWL from semantic web technology domain have more expressive power and formality of expression than OMG approaches. The properties are stand-alone entities that can exist without specific classes in case of ORLs while the properties are defined locally to a class (and its subclasses through inheritance) in case of OMG approaches like UML and MDA. For ORLs, Classes make their meaning explicit in terms of OWL statements. No imperative code can be attached. But in case of OMG approach, one needs to add data description in the form of data dictionary along with UML diagram to explain the terms. The domain models (Class Diagram) are designed as part of software architecture but the ORL uses domain models to represent knowledge about a domain, and for information integration. But UML, Java, C# etc. are much mature technologies supported by many commercial and open-source tools and expertise than the current semantic web technologies. The semantic web is an emerging technology with some open-source tools and a handful of commercial support. This makes UML and MDA as a better choice for ontology development.

7.5 Automation of Semantic Web based Digital Library

Analyzing and designing semantic web using software engineering tools is a step towards achieving automation in the whole ontology and semantic web building in general. This will help in achieving the automatic functioning and inferencing by intelligent agents. At present the commercial UML tools support automatic Java code generation in the form of templates. The programmer can fill these templates in order to produce a working code. The future plan is to work on this aspect. The automatic code generation has two important impacts on the semantic web based system implementation:

- The automatic generation of code will make possible automatic translation of models (PIMs) from MDA framework to the language of our choice (PSM). This thesis presents this transformation based on the XSLT stylesheets.
- The automatically generated Java code can be made available to the applications which can run in automatic fashion.
- The automatic OWL code generation means one can generate the semantic web ontologies easily. The work presented in chapter 4 demonstrates use of UML class models for the generation of ontologies. The UML being visual language can make ontology generation and maintenance very easy. The semantic web intelligent agents can directly make use of these generated ontologies for processing and inferencing of the query. The digital library agents can carry out tasks like classification and searching in automated fashion at runtime because of the XSLT mapping discussed in chapter 5.

Developing XSLT can be cumbersome when used for complex ontologies. There are some tools developed by UBOT and CODIP project teams for generating XSLT mappings. But these tools are not very user friendly and are not available for commercial purposes [CD2001].

7.6 Transformation of Domain Models to Code

The work presented in the thesis uses Java and OWL XSLTs to map the UML class models to the corresponding Java and OWL templates. This work has made possible use of a visual domain model like UML or MDA to model ontologies. Further these models which are easier to generate and maintain the domain ontologies in automatic manner. So the available work in the form of UML class models and Entity –Relationship models can be reused [B+2008]. There is a possibility of making use of the entire available visual domain modeling languages and models for ontology development. This can be considered a significant contribution as the present semantic web technology is still in its infancy, and not good and standard visual ontology modeling tools are available.

At present the methodology used in this thesis strongly relies on XMI representations of models. This makes it unsuitable for mapping models to OWL if they do not have XMI representations. A more generic methodology will make possible use of all the existing models for generating corresponding OWL ontologies which will further be used by intelligent agents.

8. Conclusion

This thesis has integrated work carried out by earlier researchers in the field of semantic information retrieval keeping traditional information retrieval work as the backdrop. These two ways of information retrieval have been found to differ in two major aspects; keyword/concept based search and the evaluation of relevancy of the retrieved documents. The traditional information retrieval uses recall and precision while the semantic information retrieval one more measure called uses response time for relevancy measurement. It has been found that the most successful semantic search algorithms are vector space model and the hybrid algorithm which is a combination of classical technique with spread activation algorithm. The concepts which form the basis of the semantic domain model are not orthogonal. Hence the vector model which represents semantic concepts as different axes may actually represent axes which are not orthogonal. The concepts which can be inferred from one another as depicted in the suggested modified semantic web information retrieval model of figure 4 are not independent. It has been suggested that the issue can be addressed by reassigning the weights to concept links based on the relationship graph of the ontology concepts. The information retrieval and the reassignment of weights consider only inheritance type of relationships. Semantic search systems require good domain ontology for better performance.

The semantic web based system has been modeled using software engineering based OMG’s UML technique in this thesis. The major problem faced by semantic web ontology developers is the unavailability of standard and easy to use tools for visualizing the ontologies while building them. This thesis work has addressed the problem by using OMG’s MDA framework based UML for representing domain ontologies. The MDA has made the modeling more formal and hence can make agents more capable of complex inferencing. The use of software engineering based technique has facilitated ontology development for a problem domain. This has also made quality of the information search better as UML models representing ontologies can be understood and processed by machines because of the XSLT technology. Use of other UML models like use case interaction and state models has been demonstrated for analyzing agent behavior in semantic web. The uploading and searching agent behaviour is analyzed using UML technique.

Future federation of digital libraries will be semantic web based. This thesis has presented software engineering modeling technique based design and development of domain ontologies for semantic web based digital library. It allows the domain experts to contribute significantly in the ontology development. The UML behavioral models namely interaction models and state models have been used to model the intelligent agent functionalities. As other semantic web technologies for modeling the ontologies and agents are still in infancy, the UML technique demonstrated for SW based systems can facilitate the development of semantic web based applications. The mappings from software engineering modeling languages to ontology representation languages have been worked out. This work is actually a step towards automation of SW based systems. The ontology generation can be automated and hence one needs to work on the UML class model in order to make changes to the OWL ontologies that are visible to the agents. In general, if any modeling language can have an XMI representation, the corresponding OWL code can be automatically generated using the XSLT technology used in this thesis. This thesis generates the OWL domain ontologies using XSLT transformations for the semantic web based digital library domain. This work will facilitate the reuse of existing models of complex problem domains. The use of XSLT technology can facilitate the creation and maintenance of any semantic web based system in an automated way.

9. Specific Contributions

The following are the specific contributions of this thesis:

- The semantic web based system has been modeled using software engineering modeling techniques
    - The domain knowledge for semantic web based system has been modeled using software engineering modeling techniques
    - The semantic web agent behaviour has been modeled using software engineering modeling techniques
- Application of software engineering modeling techniques for domain ontology and agent behaviour modeling of semantic web based digital library has been demonstrated.
- The effect of semantic web domain modeling techniques has been found to facilitate the semantic web information retrieval
- The XSLT mapping technique developed for semantic web based system will make possible the automation of ontology generation and maintenance.
- The XSLT transformations make possible reuse of any model of complex problem domain by semantic web based systems.
- The work on comparison of TIR and SWIR resulted in identifying an additional layer the existing SWIR model, namely, semantic concepts relationships layer.

10.References

[AMc1987] Albus, J., McCain, H., Standard Reference Model for Telerobot Control System Architecture NASREM, NBS Technical Note 1235, Robot Systems Division, NIST, 1987].

[B+2001a] Baclawski K.,Kokar M.,Kogut P.,Hart L.,Smith J.,Holmes W.,Letkowski J., Aronson M. (2001) Extending UML to support Ontology Engineering for the Semantic Web, In the Proceedings of UML 2001, Toronto, Canada, October 1-5, 2001.

[B+2001b] Baclawski K.,Kokar M.,Kogut & Aronson M. Metamodeling Facilities, working paper presented at OMG Technical Meeting, Danvers, MA, USA, July 9-13, 2001.

[B+2002] THE DESCRIPTION LOGIC HANDBOOK: Theory, implementation and applications, Cambridge University Press (Eds.)Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, Peter Patel Schneider, 2002

[Bat1988] Bates M.J., Indexing and Access for Digital Libraries and the Internet: Human, Database and Domain Factors. Journal of American Society for Information Science, 49: 1185-1205, 1988.

[Bau2003] Bauer Bernhard, Using UML in the Context of Agent-Oriented Software Engineering: State of the Art. P. Giorgini, J.P.Muller, J.Odell, (Eds): AOSE 2003, LNCS 2935, pp.1-24.

[Bau2004] Baur Bernhard, MDA Applied: From sequence Diagrams to Web Service Choreography. N.Koch, P.Fraternali (Eds.):ICWE 2004, LNCS 3140, pp. 132-136, 2004.

[BB1989] A.Borgida, R.J.Brachman. CLASSIC: A Structural Data Model for Objects, 1989 ACM SIGMOD International Conference on Management of Data, 59-67, 1989.

[Bhi2007a] Bhise Minal, Comparison of Traditional and Semantic Web Information Retrieval, First International Conference on Information Processing, ICIP 2007, 10-12 August, 2007, Bangalore, INDIA.

[Bhi2007b] Bhise, M., Semantic Web Domain Knowledge Representation Using Software Engineering Modeling Technique. ICSD, Bangalore, February 21-23, 2007.

[Bhi2008] Bhise Minal, Semantic Web Information Retrieval and Domain Knowledge Representation Using Unified Modeling Language**,** International Software Engineering Theory and Practice Conference SETP 2008, Orlando, USA , July 2008.

[Bhi2009a] Bhise Minal, Semantic Web Information Retrieval and Software Engineering Modeling Techniques, International Journal of Information Processing IJIP, ISSN 0973-8215/Volume3/ Number 1/ 2009, pp.1-11

[Bhi2009b] Bhise Minal, Automation of Semantic Web based Digital Library using Unified Modeling Language, International Journal of Recent Trends in Engineering, Finland, ISSN 1797-9617, ISBN 978-952-5726-04-6/Volume1/ Number1/2009, pp.1-5

[BHL2001] Berners-Lee T., Hendler J., Lassila O., The Semantic Web, Scientific American. 2001, 284: 35-43.

[BRJ1999] Booch G., Rumbaugh J., Jacobson I., The UML User Guide, Pearson Education, 1999

[BS1985] R.J.Branchman, J.G.Schmolze. An Overview of the KL-ONE Knowledge Representation System. Cognitive Science, 9(2): 171-216, April 1985

[BYR1999] R.Baeza-Yates, B.Ribeiro-Neto, Modern Information Retrieval, 1st edition, Addison-Wesley, 1999.

[CCF2000] Chella A., Cossentino M., Faso U.L. (2000) Applying UML Use Case Diagrams to Agents Representation, AIIA 2000, Milan, Italy

[CD2001] Cruz I., Decker S., editors. UML and the semantic web, Proceedings of 1st Semantic Web Working Symposium, California, USA; 2001, July 30-August 1, Amsterdam: IOS press; 2002.

[CN1995] H.Chen and D.T.Ng, An Algorithmic Approach to Concept Exploration in a Large Knowledge Network, Journal of American Society for Information Science, 46(5): 348-369, June 1995

[CP1999] Cranefield S. and Purvis M. UML as an Ontology Modeling Language. In the proceedings of workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence (IJCAI-99), 1999.

[Cra2001a] Cranefield, S. Networked Knowledge Representation and Exchange using UML and RDF. Journal of Digital Information. 1(8). 2001 Available from: http://jodi.tamu.edu/Articles/v01/i08/Cranefield/

[Cra2001b] Cranefield, S. UML and the Semantic Web, In Proceedings of the International Semantic Web Working Symposium SWWS, Palo Alto, July 30- August 1, 2001

[Cre1997] Crestani F, Application of Spreading Activation Techniques in Information Retrieval. Artificial Intelligence Review, 11(6), 453-482, 1997.

[Dac2003] Daconta, Michael C. (2003). *The Semantic Web*. Wiley Publishing.

[Den2005] Denny M., Ontology Tools Survey, http://www.objectsbydesign.com/tools/umltools _byCompany.html, 2005.

[DGD2004] Djuric D.,Gasevic D.,Devedzic D. Converting UML to OWL Ontologies. WWW 2004, May 17-22, 2004, NY, USA

[DGD2005] Djuric D., Gasevic D., Devedzic D.(2005). MDA for Ontology Modeling. *Journal of Object Technology*, 4(1), 109-128.

[DGD2007] Djuric D.,Gasevic D.,Devedzic D. Adopting Software Engineering Trends in AI. *IEEE Software Engineering,* 22(1), 59-66, January/ February 2007

[Dom2000] S. Dominich, Formal Foundations of Information Retrieval, in: Proceedings of the Workshop on Mathematical Methods in Information Retrieval at the International ACM SIGIR Conference on Research and Development in Information Retrieval, Athens, Greece, 2000.

[DS2004] Dean M., Schreiber G., Editors. Web Ontology Language Reference, February 2004. Available at: http://www.w3c.org/TR/OWL-ref/

[DSW2002] Dong J.S., Sun J.,Wang H. Z Approach to Semantic Web. In C.George and H.Miao, editors, International Conference on Formal Engineering Methods ICFEM 02. (pp156-167), LNCS, Springer Verlag, October 2002

[DSW2003] Dong J.S., Sun J.,Wang H. Checking and Reasoning about Semantic Web through Alloy, In Proceedings of 12th International Symposium on Formal Methods. Europe: FM'03. (pp. 796-813), Pisa Italy, LNCS, Springer Verlag, September 2003

[EKS2000] Evans A., Kent S., Selic B. editors. Processes, roles, and events: UML concepts for enterprise architecture. Proceedings of the 3rd International Unified Modeling Language Conference, York, UK; 2000, October 2-6, New York: Springer; 2000.

[Erw1996] Erway, R.L. (1996). Digital initiatives of the Research Libraries Group. *D-Lib Magazin*e, December, 1996.

[F+1988] G.W.Furnas, S.Deerwester, S.T.Dumais, T.K.Landaur, R.A.Harshman, L.A.Streeter, K.E.Lochbaum. Information Retrieval Using a Singular Value Decomposition Model of Latent Semantic Structure. 11th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.465-480, 1988

[F+1995] E.A.Fox, R.M.Akscyn, R.K.Furuta,and J.J.Leggett. Digital Libraries. Communications of the ACM, 38(4):22-28, April 1995.

[Fal2003] Falkovych K. (2003). UML for the Semantic Web: Transformation-Based Approaches. In *Knowledge Transformation for the Semantic Web, Frontiers in Artificial Intelligence & Applications*. (pp 92-106). IOS press.

[Fik1994] Genesereth M.,Fikes R.(1994). *The Knowledge Interchange Format (KIF) Reference Manual.* Stanford University, USA.

[Fre2003] Fred Waskiewicz, Editor. Catalog of OMG Modeling and Metadata Specifications, OMG Technical Report Available at: http://www.omg.org/technology/documents/modeling_spec_catalog.htm

[G+2002] Gonçalves Marcos, André Gonçalves, Edward. A. Fox, Layne T. Watson, Neill A. Kipp. Streams, Structures, Spaces, Scenarios, Societies (5S): A Formal Model for Digital Libraries. Virginia Tech CS Technical Report, TR-03-04, 2002

[GB2004] Grigoris A., Boley H., editors. Rules and Rule markup languages for the semantic web. Proceedings of the 3rd RuleML International Conference, Hiroshima, Japan; 2004, Nov 8, New York: Springer; 2004.

[GMM2003] Guha R., McCool R., Miller E., Semantic Search. Proceedings of WWW2003, May 20-24, 2003, Budapest, Hungary.

[Gru1993] Gruber T R, A Translational Approach to Portable Ontology Specifications. Knowledge Acquisition. 1993(5): 199-220

[Gua1998] N. Guarino. Formal Ontology and Information Systems. In N.Guarino(ed.). Formal Ontology in Information Systems. Proceedings of FOIS 1998, Trento, Italy, 6-8 June 1998. Amsterdam, IOS Press, pp. 3-15.

[Hen1999] James Hendler. Is there an Intelligent Agent in your Future, *Nature*, 121-129, 1999.

[HMS1996] Harum, S., Mischo, W., and Schatz, B.R., Federating Repositories of Scientific Literature: An Update on the Digital Library Initiative at the University of Illinois at Urbana-Champaign, D-Lib Magazine, July/August 1996.

[Hor2002] Ian Horrocks, Reasoning with Expressive Description Logics: Theory and Practice, University of Manchester, UK 2002.

[IVB2006] Ioana R., Valentin R., and Benoit T., An Introduction to the Semantic Web for Health Sciences Librarians, Journal of Medical Library Association, 94(2), 198-205, 2006.

[K+2000] Kalinichenko00] L. A. Kalinichenko, D. O. Briukhov, N. A. Skvortsov, and V. N. Zakharov. Infrastructure of the subject mediating environment aiming at semantic interoperability of heterogeneous digital library collections. *In Proceedings of the 2nd Russian Scientific Conference on Digital Libraries: Advanced Methods and Technologies*, Protvino. Sep. 26-28, 2000.

[K+2002] Kogut P., Cranefield S., Hart L., Dutra M., Baclawski K., Kokar M., Smith J. UML for Ontology Development. The Knowledge Engineering Review. 2002, 17(1): 61-64.

[KFW2004] Koch N., Fraternali P., Wirsing M., editors. UML profile for OWL. Proceedings of the 4th Web Engineering Conference, Munich, Germany; 2004, Jul 28-30, New York: Springer; 2004.

[KWB2003] Anneke Kleppe, Jos Warmer, Wim Bast. *MDA Explained, The Model Driven Architecture: Practice and Promise*, Addison-Wesley, 2003, ISBN 0-321-19442

[LOC1995] The Library of Congress. The National Digital Library Program. 1995 Available at: http://lcweb.loc.gov/ndl/aug-95.html

[Loo] Loom. Available at: http://www.isi.edu/isd/LOOM/LOOM-HOME.html

[Mac2003] David J C Mackay, Information Theory, Inference, and Learning Algorithms, Cambridge University Press, 2003, ISBN-13: 9780521642989

[McH1990] A.T.McCray and W.T.Hole, The Scope and Structure of the First Version of the UMLS Semantic Network. IEE Fourteenth Annual Symposium on Computer Applications in Medical Care. pp 126-130. Los Alamitos, CA, November 4-7, 1990

[MM1996] B.S.Manjunath and W.Y.Ma, Text Features for Browsing and Retrieval of Image Data. IEEE Transactions on Pattern Analysis and Machine Intelligence, 18(8): 837-841, August 1996

[NCL2006] Na H.S., Choi O.H., Lim J.E., A Method for Building Ontologies Based on The Transformation of UML Models, In Proceedings of SERA 2006, IEEE Computer Society, August 9-11, 2006

[NM2000] Noy N., McGuinness D.L., Ontology Development 101: A Guide to Creating Your First Ontology, Stanford University Technical Report, pp.4-10, 2000.

[OPB2000] Odell J, Parunak H., Bernhard B. Extending UML for Agents, Odell, J., Parunak, H. V. D. & Bauer, B.(Eds.) (2000). In Proceedings of the Agent-Oriented Information Systems Workshop at the 17th National conference on Artificial Intelligence.

[PK2003] Paralic J., Kostial I. Ontology Based Semantic Search. Proceedings of IIS 2003, Varazdin, Croatia, September 2003.

[Pok2004] Pokorny J., Web Searching and Information Retrieval, Web Engineering Volume 6, Issue 4, 43 – 48, 2004

[Por1980] Porter, M.F. An Algorithm for Suffix Stripping, Program, 14(3):130-137, 1980.

[Pro] Protege. Available at : http://protege.stanford.edu/

[PU2006] Pan J., Uschold M., A Semantic Web Primer for Object Oriented Software Developers, w3c Working Group Note, 9 March 2006

[R+1997] Rector A, Bechhofer S, Goble C, Horrocks I, Nowlan W, Solomon W. The Grail Concept Modeling Language for Medical Terminology. Artificial Intelligence Medical Journal?, 1997 Feb;9(2):139-171.

[RSA2004] Rocha C., Schwabe D., Aragao M., A Hybrid Approach for Searching in the Semantic Web. Proceedings of WWW 2004, May17-22, 2004, New York, USA.

[S+1999] B.R.Schatz, B. Mischo, T.Cole, a.Bishop, S. Harum, L.Neumann, H.Chen and T.D.Ng. Federating Search of Scientific Literature. IEEE Computer, 32 (2): 51-59, February 1999

[SB1988] Gerard Salton, Chris Buckley, Term-Weighting Approaches in Automatic Text Retrieval. Inf. Process. Manage. 24(5): 513-523 (1988)

[SF2002] Suleman, H., Fox, E.A., "The Open Archives Initiative: realizing simple and effective digital library interoperability", *Journal of Library Automation*, Vol. 35 No.1/2, pp.122-45, 2002

[SFJ2002] Urvi Shah, Tim Finnin, Anupam Joshi., Information Retrieval on the Semantic Web, CIKM'02, 2002, Virginia, USA, pp 461-468.

[SHB2006] Shadbolt N., Hall W., and Berners-Lee T., The Semantic Web Revisited, IEEE Intelligent Systems, 21 (3), 96-101, 2006.

[SpL2002] SpeedLegal Inc. Xerlin Project. Version 1.0. URL: http://www.xerlin.org/. 2002.

[W+1996] H.D. Wactlar, T.Kanade, M.A.Smith, and S.M. Stevens. Intelligent Access to Digital Video: Informedia Project. IEEE Computer, 29(5):46-53, May 1996

[Wan1999] Bing Wang. A hybrid system approach for supporting digital libraries. *International Journal on Digital Libraries*, 2(2-3):91-110, 1999.

[Wil1996] R.Wilensky, Toward Work-Centered Digital Information Services. IEEE Computer, 29(5): 37-45, May1996

[You1997] Younger, J. (1997). Resources description in the digital age. Dublin Core: basic set of data elements. *Library Trends, 45*(3), 462-481.

[Z+2003] Zhu Q., Goncalves M.A., Shen R., Fox E.A., Visual Semantic Modeling of Digital Libraries. T.Koch and I.T.Solvberg (Eds.): ECDL 2003, LNCS 2769, pp.325-337, 2003.

Appendix A

Java Code for Semantic Web based Digital Library Domain Ontology using Rational Rose Tool

<u>Classes</u>

1. **Resource:**

```
//Source code: Resource.java
     public class Resource
{
   private String RscName;
   private Integer RscID;
   private String RscLanguage;
   private String RscLink;
   private Boolean Readable;
   private Boolean Downloadable;
   private String RscSubject;
   private String RscUploader;
   private String RscOverview;

   /**
   @roseuid 4896E15901B9
    */
   public Resource()
   {

   }
}
```

1.1 Journals: Extends from Resource Class

```
   //Source code: Journals.java
 public class Journals extends Resource
  {
       private Object Author;

   /**
   @roseuid 4896E159038E
    */
   public Journals()
   {

   }
}
```

1.2 Ebook: Extends from Resource Class

```
//Source code: Ebook.java
public class Ebook extends Resource
{
   private Object Author;
   private Object Publisher;
   private String Edition;

   /**
   @roseuid 4896E15A0042
    */
   public Ebook()
   {

   }
}
```

1.3 Conference Paper: Extends from Resource Class

```
      //Source code:ConferencePaper.java
public class ConferencePaper extends Resource
{
   private String Editor;
   private Integer ConfD;
   private Object Authors;

   /**
   @roseuid 4896E15A00DE
    */
   public ConferencePaper()
   {

   }
}
```

1.3.1 Conference: Extends from Conference Paper Class

```
//Source code: Conference.java
public class Conference extends ConferencePaper
{
   private Object Oraganization;
   private String Venue;
   private Date Date;
   private Integer ConferenceID;
   private String ConferenceSubject;
   private String ConferenceOverview;

   /**
   @roseuid 4896E15A01A9
```

```
     */
    public Conference()
    {

    }
 }
```

1.4 Reports: Extends from Resource Class

```
      //Source code: Reports.java
 public class Reports extends Resource
 {
    private Date Date;
    private Object Author;
    private Object Organization;

    /**
    @roseuid 4896E15902E2
     */
    public Reports()
    {

    }
 }
```

1.5 Project: Extends from Resource Class

```
 //Source code: Project.java
 public class Project extends Resource
 {
    private String Developers;
    private String Developer_Email;
       /**
    @roseuid 4896E15A036E
```

```
     */
    public Project()
    {

    }
 }
```

1.6 Article: Extends from Resource Class

```
 //Source code: Article.java
 public class Article extends Resource
 {
    private Object Author;
    private Object Publisher;

    /**
    @roseuid 4896E15B0032
     */
    public Article()
    {

    }
 }
```

1.7 MultiMedia: Extends from Resource Class

```
 //Source code: Multimedia.java
 public class Multimedia extends Resource
 {
    private String Type;
    private String Knowledge_Provider;

    /**
```

```
   @roseuid 4896E15B00CF
    */
   public Multimedia()
   {

   }
}
```

2 Organization: Association with Conference and Reports Class

```
//Source code: Organization.java
public class Organization
{
   private String HeadOffice;
   private Date StartedOn;
   private String Overview;

   /**
   @roseuid 4896E15A0274
    */
   public Organization()
   {

   }
}
```

3 Subject: Association with Resource and Librarian Class

```
//Source code: Subject.java
public class Subject
{
   private Integer ID;
   private String Info;
   private String Name;
   /**
   @roseuid 4896E15B016B
    */
   public Subject()
   {

   }
}
```

4 Person:

```
//Source code: Person.java
public class Person
{
   private String Name;
   private String Email;

   /**
   @roseuid 4896E15801C8
    */
   public Person()
   {

   }
}
```

4.1 Author: Extends from Person Class

```
//Source code: Author.java
public class Author extends Person
{
   private String Biography;
   private String Interests;
   /**
   @roseuid 4896E15802B3
    */
   public Author()
   {

   }
 }
```

4.2 Publisher: Extends from Person Class

```
    //Source code: Publisher.java
public class Publisher extends Person
{
   private String HeadOffice;
   private Date StartingDate;

   /**
   @roseuid 4896E158034F
    */
   public Publisher()
   {

   }
 }
```

4.3 LibraryMemeber: Extends from Person Class

```
//Source code: LibraryMember.java
public class LibraryMember extends Person
{
   private Integer ID;
   private String Password;
```

```
  /**
  @roseuid 4896E1590003
   */
  public LibraryMember()
  {

  }
}
```

4.3.1 Librarian: Extends from LibraryMember Class

```
//Source code: Librarian.java
public class Librarian extends LibraryMember
{

  /**
  @roseuid 4896E1590090
   */
  public Librarian()
  {

  }
}
```

4.3.2 EndUser: Extends from LibraryMember Class

```
//Source code: EndUser.java
public class EndUser extends LibraryMember
{
  private String ProfileLink;

  /**
  @roseuid 4896E159011D
```

```
    */
   public EndUser()
   {

   }
}
```

4.3.2.1 Professor: Extends from EndUser Class

```
//Source code: Professor.java
public class Professor extends EndUser
{
   private String University;
   private String AreaofSpecilalization;

   /**
   @roseuid 4896E15B0311
    */
   public Professor()
   {

   }
}
```

4.3.2.2 Students: Extends from EndUser Class

```
//Source code : Students.java
public class Students extends EndUser
{
   private String Interest;

   /**
   @roseuid 4896E15B03CC
    */
   public Students()
   {

   }
}
```

4.3.2.3 Others: Extends from EndUser Class

```
//Source code: Others.java
public class Others extends EndUser
{

   /**
   @roseuid 4896E15C0061
    */
   public Others()
   {

   }
}
```

5. **Comments: Association with EndUser Class**

```
 //Source code: Comments.java
public class Comments
{
   private Integer User_ID;
   private String Content;
   private Integer RscID;
   private Integer CommentID;

   /**
   @roseuid 4896E15C00EE
```

```
     */
    public Comments()
    {

    }
 }
```

6 Request: Association with EndUser Class

```
  //Source code: Request.java
 public class Request
 {
    private Integer Rqst_ID;
    private String Rqst_Name;
    private Boolean Rqst_Status;
    private Integer Rqst_UserID;
       /**
    @roseuid 4896E15C018A
     */
    public Request()
    {

    }
 }
```

7 DigitalLibraryInstitute: Association with Librarian Class

```
 //Source code: DigitalLibraryInstitute.java
 public class DigitalLibraryInstitute
 {
    private String Name;
    private Integer ID;
    private String WebsiteLink;
    private String Info;
```

```
     /**
   @roseuid 4896E15B0226
    */
   public DigitalLibraryInstitute()
   {

   }
}
```

Appendix B

OWL Code Generated for Semantic Web based Digital Library Domain Ontology using XSLT Mapping

```
<?xml version="1.0"?>


<!DOCTYPE rdf:RDF [
  <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
  <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>


<rdf:RDF xmlns="http://www.owl-ontologies.com/Ontology1205834915.owl#"
   xml:base="http://www.owl-ontologies.com/Ontology1205834915.owl"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:owl="http://www.w3.org/2002/07/owl#">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="Articles">
    <rdfs:subClassOf rdf:resource="#Resource"/>
    <owl:disjointWith rdf:resource="#Journal"/>
    <owl:disjointWith rdf:resource="#E-Book"/>
    <owl:disjointWith rdf:resource="#Conference_Paper"/>
    <owl:disjointWith rdf:resource="#Reports"/>
    <owl:disjointWith rdf:resource="#Projects"/>
    <owl:disjointWith rdf:resource="#Multimedia"/>
  </owl:Class>
  <owl:Class rdf:ID="Audio_files">
    <rdfs:subClassOf rdf:resource="#Multimedia"/>
    <owl:disjointWith rdf:resource="#Video_files"/>
```

```
</owl:Class>
<owl:DatatypeProperty rdf:ID="canRead">
    <rdfs:subPropertyOf rdf:resource="#hasRights"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="canReadandDownload">
    <rdfs:subPropertyOf rdf:resource="#hasRights"/>
</owl:DatatypeProperty>
<owl:Class rdf:ID="Conference"/>
<owl:Class rdf:ID="Conference_Paper">
    <rdfs:subClassOf rdf:resource="#Resource"/>
    <owl:disjointWith rdf:resource="#Journal"/>
    <owl:disjointWith rdf:resource="#E-Book"/>
    <owl:disjointWith rdf:resource="#Reports"/>
    <owl:disjointWith rdf:resource="#Projects"/>
    <owl:disjointWith rdf:resource="#Articles"/>
    <owl:disjointWith rdf:resource="#Multimedia"/>
</owl:Class>
<owl:Class rdf:ID="E-Book">
    <rdfs:subClassOf rdf:resource="#Resource"/>
    <owl:disjointWith rdf:resource="#Journal"/>
    <owl:disjointWith rdf:resource="#Conference_Paper"/>
    <owl:disjointWith rdf:resource="#Reports"/>
    <owl:disjointWith rdf:resource="#Projects"/>
    <owl:disjointWith rdf:resource="#Articles"/>
    <owl:disjointWith rdf:resource="#Multimedia"/>
</owl:Class>
<owl:DatatypeProperty rdf:ID="hasAuthor">
    <rdfs:domain>
        <owl:Class>
            <owl:unionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#Articles"/>
                <owl:Class rdf:about="#Conference_Paper"/>
                <owl:Class rdf:about="#E-Book"/>
                <owl:Class rdf:about="#Journal"/>
                <owl:Class rdf:about="#Multimedia"/>
                <owl:Class rdf:about="&owl;Thing"/>
            </owl:unionOf>
        </owl:Class>
    </rdfs:domain>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="hasConfID">
    <rdfs:domain rdf:resource="#Conference"/>
```

```
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="hasDate">
  <rdfs:domain rdf:resource="#Reports"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="hasDevelopers">
  <rdfs:domain rdf:resource="#Projects"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="hasEdition">
  <rdfs:domain rdf:resource="#E-Book"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="hasEditors">
  <rdfs:domain rdf:resource="#Conference_Paper"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="hasLanguage">
  <rdfs:domain rdf:resource="#Resource"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="hasLink">
  <rdfs:domain rdf:resource="#Resource"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="hasName">
  <rdfs:domain rdf:resource="#Resource"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="hasOrganization">
  <rdfs:domain>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Conference"/>
        <owl:Class rdf:about="#Reports"/>
      </owl:unionOf>
    </owl:Class>
  </rdfs:domain>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="hasOverview">
  <rdfs:domain rdf:resource="#Resource"/>
</owl:DatatypeProperty>
<owl:ObjectProperty rdf:ID="hasPublisher">
  <rdfs:domain>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Articles"/>
        <owl:Class rdf:about="#E-Book"/>
      </owl:unionOf>
```

```
        </owl:Class>
    </rdfs:domain>
    <rdfs:range rdf:resource="#Publisher"/>
</owl:ObjectProperty>
<owl:DatatypeProperty rdf:ID="hasRights">
    <rdfs:domain rdf:resource="#Resource"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="hasRscID">
    <rdfs:domain rdf:resource="#Resource"/>
</owl:DatatypeProperty>
<owl:ObjectProperty rdf:ID="hasSubject">
    <rdfs:domain>
        <owl:Class>
            <owl:unionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#Conference"/>
                <owl:Class rdf:about="#Resource"/>
            </owl:unionOf>
        </owl:Class>
    </rdfs:domain>
    <rdfs:range rdf:resource="#Subject"/>
</owl:ObjectProperty>
<owl:DatatypeProperty rdf:ID="hasUploader">
    <rdfs:domain rdf:resource="#Resource"/>
</owl:DatatypeProperty>
<owl:Class rdf:ID="Journal">
    <rdfs:subClassOf rdf:resource="#Resource"/>
    <owl:disjointWith rdf:resource="#Projects"/>
    <owl:disjointWith rdf:resource="#Multimedia"/>
    <owl:disjointWith rdf:resource="#Reports"/>
    <owl:disjointWith rdf:resource="#E-Book"/>
    <owl:disjointWith rdf:resource="#Articles"/>
    <owl:disjointWith rdf:resource="#Conference_Paper"/>
</owl:Class>
<owl:Class rdf:ID="Multimedia">
    <rdfs:subClassOf rdf:resource="#Resource"/>
    <owl:disjointWith rdf:resource="#Journal"/>
    <owl:disjointWith rdf:resource="#E-Book"/>
    <owl:disjointWith rdf:resource="#Conference_Paper"/>
    <owl:disjointWith rdf:resource="#Reports"/>
    <owl:disjointWith rdf:resource="#Projects"/>
    <owl:disjointWith rdf:resource="#Articles"/>
</owl:Class>
```

```
  <owl:Class rdf:ID="Projects">
    <rdfs:subClassOf rdf:resource="#Resource"/>
    <owl:disjointWith rdf:resource="#Journal"/>
    <owl:disjointWith rdf:resource="#E-Book"/>
    <owl:disjointWith rdf:resource="#Conference_Paper"/>
    <owl:disjointWith rdf:resource="#Reports"/>
    <owl:disjointWith rdf:resource="#Articles"/>
    <owl:disjointWith rdf:resource="#Multimedia"/>
  </owl:Class>
  <owl:Class rdf:ID="Publisher"/>
  <owl:Class rdf:ID="Reports">
    <rdfs:subClassOf rdf:resource="#Resource"/>
    <owl:disjointWith rdf:resource="#Journal"/>
    <owl:disjointWith rdf:resource="#E-Book"/>
    <owl:disjointWith rdf:resource="#Conference_Paper"/>
    <owl:disjointWith rdf:resource="#Projects"/>
    <owl:disjointWith rdf:resource="#Articles"/>
    <owl:disjointWith rdf:resource="#Multimedia"/>
  </owl:Class>
  <owl:Class rdf:ID="Resource"/>
  <owl:Class rdf:ID="Subject"/>
  <owl:Class rdf:ID="Video_files">
    <rdfs:subClassOf rdf:resource="#Multimedia"/>
    <owl:disjointWith rdf:resource="#Audio_files"/>
  </owl:Class>
</rdf:RDF>
```

Publications and Presentations

Publications

International Conference Publications

Bhise M., *Software Engineering Modeling Techniques and Semantic Web,* International Conference on Software Engineering Theory and Practice, **SETP 2008**, 7-10 July, 2008, Orlando, USA

Bhise M., *Comparison of Traditional and Semantic Web Information Retrieval",* First International Conference on Information Processing, **ICIP 2007**, 10-12 August, 2007, Bangalore, INDIA.

Bhise M., *Semantic Web: Domain Ontology Representation Using Software Engineering Modeling Technique and Information Retrieval,* Conference on Information Science Technology and Management, **CISTM 2007**, 16-18 July, 2007, Hyderabad, INDIA

Bhise M., *Semantic Web Domain Knowledge Modeling Using Software Engineering Modeling Technique,* International Conference on Semantic Web and Digital Libraries **ICSD 2007**, 21-23 February, 2007, Bangalore, INDIA

Bhise M., *Semantic Web Domain Knowledge Representation and Information Retrieval*, International Workshop on Research Issues in Digital Library **IWRIDL 2006**, December 2006, Kolkata, INDIA (Position Paper)

Journal Publications

Bhise M., *Semantic Web Information Retrieval and Software Engineering Modeling Techniques*, International Journal of Information Processing IJIP, ISSN 0973-8215/Volume3/ Number 1/ 2009, pp.1-11

Bhise M., *Automation of Semantic Web based Digital Library using Unified Modeling Language*, International Journal of Recent Trends in Engineering, Finland, ISSN 1797-9617, ISBN 978-952-5726-04-6/Volume1/ Number 2 /2009, pp.205-209

Bhise M., *Domain Ontology Modeling for the Semantic Web Based Digital Library*, IJKM, IGI Global, USA (Communicated)

Presentations at International Conferences

Bhise M., *Comparison of Traditional and Semantic Web Information Retrieval,* International Conference on Information Processing, **ICIP 2007**, August 10-13, Bangalore, INDIA

Bhise M., *Semantic Web: Domain Ontology Representation Using Software Engineering Modeling Technique and Information Retrieval,* Conference on Information Science Technology and Management, **CISTM 2007**, July 16-18, Hyderabad, INDIA

Bhise M., *Semantic Web Domain Knowledge Modeling Using Software Engineering Modeling Technique,* International Conference on Semantic Web and Digital Libraries **ICSD 2007**, February 21-23, Bangalore, INDIA

Brief Biography of the Candidate

Personal Details

Name	Minal Bhise
Address	1209, Faculty Block 1, DAIICT, Gandhinagar, Gujarat, India 382007.
Date of Birth	14 September, 1968

Education

Ph.D.	(Computer Science)	BITS, Pilani (Submission 2009)
M.E.	(Software Systems)	BITS, Pilani, 1999. CPI 9.0
M.Sc.	(Physics)	University of Pune, 1990, CPI 9.3,
B.Sc.	(Physics)	University of Pune, 1988, CPI 9.5

Work Experience

2003 -	Assistant Professor, DAIICT, Gandhinagar
1997-2002	Lecturer, Computer Science Department, BITS, Pilani
1994-1996	Scientist-C, Institute for Plasma Research, Bhat, Gandhinagar
1991-1993	Research Scholar, Institute of Plasma Research, Bhat, Gandhinagar

Research Interests

Multimedia Databases, Information Retrieval, Knowledge Management using Software Engineering Modeling Techniques, Semantic Web, Intelligent Agents

PhD Thesis (2000-2009)

Software Engineering Modeling Techniques for Semantic Web based Digital Library

Supervisor: Prof. Prabhat Ranjan, DAIICT, Gandhinagar

Physics Publications 1991-1995

Bhise M.K.et.al., *Design Point Selection for a Superconducting Steady State Tokamak*, IAEA Technical Committee Meeting on Research Using Small Tokomaks, Madrid, Spain, September 22-23, 1994.

Bhise M.K. et.al. *Design of Vacuum System and Support Structure for Plasma Facing Components of SST-1 Tokomak*, Fusion Technology, 805-808, 1996.

Bhise M.K.et.al., Conceptual Design of Gas Feed System for SST-1, SST1/CD-GFS/003, September 1995, Institute for Plasma Research, Gandhinagar, Technical Report.

Other Professional Activities during the period of Thesis Work

Sessions Chaired at International Conferences

CISTM 2007, Hyderabad, India on *System Design and Specification*

External Reviewer

International Conferences

International Conference on Enterprise Information Systems, ICEIS 2009, Milan, Italy, May 2009

International Conference on Semantic E-Business and Enterprise Computing SEEC 2008, September 2008

International Conference on Information Science and Technology, CISTM 2008, Delhi, India, 7-9 August 2008

International Conference on Software Engineering Theory and Practice, SETP 2008, Orlando, USA, July 2008

International Journals

Interdisciplinary Journal of Information, Knowledge, and Management IJIKM, USA

International Journal of Knowledge and Information Systems KAIS, Springer.

Information and Software Technology Journal, Elsevier, Netherlands.

International Program Committee Member

International Conference on Enterprise Information Systems, ICEIS 2009, Milan, Italy, May 2009

International Conference on Semantic E-Business and Enterprise Computing SEEC 2008, India, September 2008

International Conference on Information Science and Technology, CISTM 2008, Delhi, India, 7-9 August 2008

International Conference on Software Engineering Theory and Practice, SETP 2008, Orlando, USA, July 2008

Brief Biography of the Supervisor

Name:	Prof Prabhat Ranjan
Qualification:	M.Sc. Ph.D. (University of California, Berkley)
Designation and Organization:	Professor, DAIICT, Gandhinagar
Area of Research:	Computer Science, Sensor Networks, Nuclear Fusion
Work Experience (years):	25
Number of Publications:	65
PhD Students Supervised:	03