Abstract— The semantic web ontology building process is time consuming and difficult and needs domain experts to define basic concepts and structures. There are no standard visual ontology building tools available which can map ontologies to machine understandable representations which will be used by the intelligent agents for inferencing and integration. The Unified Modeling Language (UML) being a standard visual modeling language in software engineering, it is better supported in terms of expertise and the tools as compared to the upcoming Web Ontology Language, OWL. For semantic web based digital library, development of good domain ontologies is very crucial to achieve interoperability among various components. This paper demonstrates use of UML domain modeling techniques to domain ontology using rational rose tool for digital library case study. The Extensible Stylesheet Language (XSLT) mappings have been used to map the UML ontology model to the OWL code automatically which can be used by intelligent agents to handle complex semantic queries for digital libraries. This technique would automate the ontology generation and maintenance for digital library. It also helps in automation of the inferencing and complex query processing by semantic web agents. This technique would make possible use of all the available visual domain modeling languages and models for ontology development.

Index Terms— semantic web, digital library automation, XSLT mapping, OWL, rational rose tool

I. INTRODUCTION

The semantic web based systems of future will be more scalable, flexible, extensible, and interoperable as compared to the present day web based systems. These future systems will be consisting of smaller, independent systems, each providing access to different contents are expected to work together, and hence interoperability among the smaller systems is necessary. These smaller systems can work together if they are supported by ontology. The semantic web intelligent agents and ontologies will make such web based systems more scalable, flexible, extensible, and interoperable. 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 [1]. 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 various Ontology Representation Languages (ORLs) like Knowledge Interchange Format (KIF), Simple HTML Ontology Extensions (SHOE), Topic Maps, OWL lack good visual modeling tools which are a must for human comprehension of ontologies [2]. 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. 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 complexity of the artificial intelligence based ontology building tools and lack of good, standard interfaces for ontology builders make the ontology building very difficult for the domain experts. Distinct from traditional libraries, digital libraries process large collections of digital objects and provide on-line information services. 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. If we want to make digital library available on the semantic web, we have 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. The ontology building tools which have been used for digital libraries lack good visual modeling interfaces. The work reported by Goncalves has proposed 5S (Streams, Structures, Spaces, Scenarios, and Societies) theory [3]. This approach was a bit too complicated and is not very easy for the domain experts to follow for building digital library [4].
II. DOMAIN ONTOLOGY MODELING USING UNIFIED MODELING LANGUAGE

The best possible ontology solution 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 your 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. 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.

UML is a modeling language defined by Object Management (OMG) group under its modeling standard Model Driven Architecture (MDA) [5]. 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. This truncated list of nouns is the list of classes. This list has to be analyzed for what each class is supposed to know and assign attributes to them. The possible functionalities of the classes in the form of use cases and their scenarios have been investigated in the presented work. By walking through the scenarios which is a set of actions responsible for that use case scenario, the relationships among classes have been found out. For ontology modeling purpose we need to limit ourselves to the list of classes and their hierarchies.

The way UML class models are built fits well into the general ontology building process [6,7]. The Table 1 shows comparable modeling features of UML and OWL languages which make possible use of UML for ontology building.

III. MAPPINGS FROM UML CLASS MODEL TO CODE

The XML Model Interchange Language (XMI) defines a standard way to serialize the UML diagrams. UML classes can also be mapped to sets of Java classes and RDF [8,9]. The domain experts create an ontology using ontology editor and the graphical representation of ontology using UML tool like Rational Rose.
The XMI files are created as shown in Fig. 1. A pair of XSLT then creates Java files and ORL representation of ontology respectively. The Java classes can be used by applications for representing knowledge as in memory data structures. The ORL representation can be used for domain specific information. 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 [8,10]. The UML models are exported using Extensible Language Model Interchange XMI format. The XMI format serves as input to the transformation process along with the XSLT transformation rules. Using XSLT for Java and OWL, Java and OWL specification can be obtained which can be understood and used by semantic web intelligent agents directly. Both UML and OWL can be serialized in XMI format so that the transformation from UML to OWL using XSLT can be done.

Since XMI and OWL use XML syntax, the structure and content of a XMI document can be converted 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 work uses the templates defined by Cranefield [2] and Djuric [11] which is based on Baclawaski’s transformations [12].

The UML classes and interfaces can be mapped 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 [2] has been used by us.

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. The OWL code can be generated from the UML class model using the transformations defined in Djuric’s work [11] which is based on Baclawaski’s transformations [12]. The UML models can be exported to XMI format. The transformations exploit the fact that both XMI and OWL are encoded using XML. UML models can be exported to XMI format. The UML models are exported to XMI format. The XMI format serves as input to the transformation process. XSLT converts UML models to OWL and Java codes.

```
public class Author extends Person {
    /**
     * Biography of the author.
     */
    private String biography;

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

    public Author() {
    }
}
```

Figure 3. Java Code Fragment for Author Hierarchy using Rational Rose
IV. DIGITAL LIBRARY DOMAIN ONTOLOGY MODEL USING UML

The identified concepts (ontological classes) represented as UML classes from the problem domain are administrator, user, library, author, digital resource, publisher etc. The class digital resource can be further specialized into subclasses like books, research papers, multimedia, conference publication, and digital image. The class Person can be specialized into subclasses like library member, author, and publisher. The domain model of digital library system using UML class diagram has been developed using UML rational rose tool. Author Hierarchy from the developed ontology has been depicted in Fig. 2. Some of the attributes and methods of identified classes have been shown in it.

A. Java Code Generation from Digital Library Ontology Model

The rational rose tool generates java templates for the input UML class model. The Java code that has been generated for the Author hierarchy of Fig. 2 using rational rose tool is shown in Fig. 3. Here Author is a subclass of superclass Person.

B. OWL Code Generation from Digital Library Ontology Model

The rational rose tool has been used to develop the domain ontology model for the semantic web based digital library. The generated .mdl domain 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. Using the XMI code and XSLT for OWL, OWL code for the model has been generated and is shown in Fig. 4.

V. DISCUSSION

Analyzing and designing semantic web using software engineering tools is a step towards achieving automation in the 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 the automatic java code generation in the form of template generation. The programmer can fill these templates in order to generate 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. Firstly, the automatic generation of codes will make possible automatic translation of platform independent models (PIMs) from MDA framework to the platform specific model (PSM) of our choice. 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. Secondly, the automatic OWL code generation means one can generate the semantic web ontologies easily. The work presented in this paper demonstrates use of UML class models for the generation of OWL 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 in OWL 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 this paper. The XSLT based technique in general can be used to map any existing models to the corresponding OWL and Java code if they have XMI representations.

VI. CONCLUSION

The domain ontology for semantic web based system has been modeled using software engineering modeling techniques. The XSLT mapping technique developed for semantic web based digital library automates ontology generation and maintenance. One needs to work only on the UML class model in order to make changes to the OWL ontologies that are visible to
the intelligent agents. The presented work uses java and OWL XSLT mappings to map the UML class models to the corresponding java and OWL templates. This work has made possible use of a visual domain models like UML and MDA to model ontologies. Further these models can be easily used to generate and maintain the domain ontologies in automatic manner. So the available models for the complex real life domains in the form of UML class models and Entity – Relationship models can be reused directly. There is a possibility of making use of all the available visual domain modeling languages and models for ontology development. This is a significant contribution as the present semantic web technology is still in its infancy and standard visual ontology modeling tools are available. Analyzing and designing semantic web using software engineering tools is a step towards achieving automation in the ontology and semantic web building in general. This will help in achieving the automatic functioning and inferencing by intelligent agents.

REFERENCES