CHAPTER 1

INTRODUCTION

1.1 GENERAL

This chapter introduces the background of the research to be presented in this thesis. It starts with an introduction to the general area of Semantic web, web resources and discusses the motivation and challenges for representing these web resources to add semantics. This chapter also presents a short overview of Resource Description Framework, Ontology and the need for knowledge representation and enhancing search. It also presents the survey about modeling, web pages, wiki pages and web services.

1.2 SEMANTIC WEB

The World Wide Web (WWW) is progressing from machine readable form to machine understandable form. This form helps for easy retrieval of data by software applications and computer agents by describing data in machine understandable format. Semantic web is evolved to make computer understand the document content and relationship between them. The term “Semantic Web” coined by Tim Berners-Lee et al (2001) is not a separate web, but an extension which aims to create metadata–rich web of resources, to give well-defined meaning and to enable man and machine to work in cooperation. Semantic web describes web resources and relationship between these resources using Resource Description Framework (RDF). It tries to move from the principle of web of documents to web of data. It
extends web by annotating the web resources with semantic description thereby allows machines to retrieve and process the information automatically.

Semantic web is a collection of resources like documents, images, data, information that are integrated by semantics. The semantics are inferred by defined conceptualization, inference rules and ontologies. It aids computers to manipulate information based on meaning thereby supports web automation, integration with web services, information integration and interoperability. Ontologies can be developed to describe concepts in a specific domain which facilitates sharing of significant information. Semantic properties and relationship is set between different web resources to enable computers manipulate and search automatically. The various issues related to semantic web is stated by Tim Berners-Lee (1998).

The challenges in semantic web are content maintenance, multilinguality, visualization, scalability, constructing and reusing ontologies. Content maintenance is that the changes made in the existing web content should also be automatically updated in the semantic web content. Multilinguality is supported when the information can be accessed in the native language. Visualization is needed to understand the information in visual form. In semantic web query should be dynamically produced in search engine and hyperlink for query should be shown in scalable manner. Also, constructing and reusing Web ontologies is a key challenge in semantic web.

1.3 WEB RESOURCES

Web Page is a collection of information, consisting of one or more Web resources. A web resource is identified by Uniform Resource Identifier (URI). A URI specification describes a resource as “anything that has identity”. Examples are web page, collection of web pages, service and
collection of other resources as stated in Web resources (2012) and Tim Berners-Lee (2009). In the semantic web, abstract resources and their semantic properties are described using the family of languages based on Resource Description Framework (RDF). RDF is used to describe resources i.e. metadata of resources. A RDF description of a resource is a set of triples (subject, predicate, object) where subject represents the resource to be described, predicate a type of property relevant to this resource, whereas object can be data or another resource. The predicate itself is considered as a resource and identified by a URI. RDFS and Web Ontology Language (OWL) can also be used to describe resources.

Some of the abstract resources such as classes, properties, concepts are all identified by URIs. Resources are "defined" by formal descriptions which anyone can publish, copy and modify over the Web. Web resource can be owned by its publisher, but abstract resource can be defined by an accumulation of RDF descriptions that may not be owned by someone.

1.4 RDF

Resource Description Framework (RDF) is a framework to define URI in a triple-based representation which has subject, predicate and object. It is used to exchange information about resources. These triples can be represented by URIs. Hence separate resources at different locations can be linked via the URIs. RDF schema describes the relations between classes, types, objects and data which define the usages of RDF. Uniform Resource Identifiers (URI) are global resource identifiers which are independent of arbitrary syntaxes or vocabularies.

RDF is a general-purpose language for representing information in the Web. The broad goal of RDF is to define a mechanism for describing resources. The extension is *.rdf and the registered mime type should be
"application/rdf+xml". RDF is the foundation for processing metadata. It enables the creation and exchange of resource metadata as any other Web data. RDF provides representation, schema definition and Extensible Markup Language (XML) syntax. Representation has metadata based on directed labeled graphs in which nodes are called resources (or literals) and edges are called properties. So, it is called Graph Data Model which is shown in Figure 1.1. This is also defined by W3C recommendation (2004). Schema Definition Language is to create vocabularies of labels for these graph nodes (called classes) and edges (called property types). XML syntax is for expressing metadata and schemas in a form that is both human readable and machine understandable as stated by W3C consortium (1999).

![Figure 1.1 Graph Data Model](image)

RDF is very useful in resource discovery. It provides better search engine capabilities, catalogs for describing the content and content relationships that is available at a particular Web site, page, or digital library. It also assists intelligent software agents to share knowledge, rate content and describe collection of pages that represent a single logical “document”. RDF is also used to describe intellectual property rights of Web pages, and to express privacy preferences of a user as well as the privacy policies of a Web site.

### 1.5 Ontology

Ontologies are definition of domain knowledge which organizes all entities described in a group of database schemas. Ontologies can be generated by extracting knowledge from various data resources such as
textual data, dictionary, knowledge base, database schema, and Unified Modeling Language (UML) models. Web Ontology Language (OWL) is one of the languages for creating Ontology which is given in OWL reference (2004). The created OWL for the particular application is stored in the repository.

When the user gives the web request, the web service will refer and invoke relevant ontology in the OWL Repository and access the data automatically.

<table>
<thead>
<tr>
<th>Class ontology</th>
<th>hasNonFunctionalProperties type nonFunctionalProperties</th>
</tr>
</thead>
<tbody>
<tr>
<td>importsOntology type</td>
<td>ontology</td>
</tr>
<tr>
<td>usesMediator type</td>
<td>ooMediator</td>
</tr>
<tr>
<td>hasConcept type</td>
<td>concept</td>
</tr>
<tr>
<td>hasRelation type</td>
<td>relation</td>
</tr>
<tr>
<td>hasFunction type</td>
<td>function</td>
</tr>
<tr>
<td>hasInstance type</td>
<td>instance</td>
</tr>
<tr>
<td>hasAxiom type</td>
<td>axiom</td>
</tr>
</tbody>
</table>

**Figure1.2 Sample Ontology Definition**

Ontology defines concepts and relationships used to describe and represent an area of knowledge as shown in Figure 1.2. Ontology is a formal explicit description of concepts in a domain of discourse (class), properties of each concept describing various features and attributes of the concept (properties). Ontologies are used to classify the terms used in a particular application, characterize possible relationships, and define possible constraints on using those relationships. Developing an ontology includes: defining classes in the ontology, arranging the classes in a taxonomic (subclass–super class) hierarchy, defining slots and describing allowed values
for these slots and filling in the values for slots for instances. Ontology mapping is discussed by Li Xueyong et al (2010) and query rewriting using ontology is done by Mahmoud et al (2010). Ontologies help in data integration though ambiguity exists but help to discover new relationships.

1.6 ONTOLOGY IN SOFTWARE ENGINEERING

Software engineering gives a systematic, quantifiable approach for software development. Ontologies are useful throughout the life cycle. Ontologies are used to describe requirements and to represent these requirements as knowledge. It acts as a reusable components repository, easy cross reference in coding phase, conceptual modeling and knowledge representation mapping to share semantics, to check interdependencies and to add semantics to business rules and test cases. Ontology-Driven Software Engineering (ODiSE) refers to different ways in which Ontologies contribute to capture real world domains- its processes and its artifacts. Ontological domain models can drive or refine typical development phases, such as requirements, design and implementation. There is a need for robust and scalable solutions.

1.7 SEMANTIC ANNOTATION

Semantic Annotation is tagging the entities in the text and links with their semantic descriptions. Resources are described by annotating it with metadata. It creates relationship between web resources thereby aids in data sharing and integration. Barnard (2003), Jinhyn Ahnet al (2008) and Semantic Annotation (2011) describe about the need for annotation and the usefulness in several applications. Annotations are possible using HTML5 microdata and SAWSDL.
1.7.1 HTML5 Micro Data

HTML5 core language of WWW introduces a number of semantic elements which include: <section>, <nav>, <article>, <aside>, <hgroup>, <header>, <footer>, <time> and <mark> as discussed by Mark Pilgrim (2010). <div>tag is introduced to bring semantics. The HTML5 introduces microdata, a way to annotate web pages with semantic metadata using attributes rather than separate XML documents. Microdata vocabulary defines a set of named properties. The semantic elements tell the browser and web crawlers clearly the type of content contained within the element. The major advantage of Microdata is its interoperability, i.e. any RDF representation of ontology can be mapped to HTML5 microdata.

1.7.2 SAWSDL

Semantic Annotations for Web Services Description Language (SAWSDL) is the technology standardized by W3C recommendations (2007) for Semantic Web Services. SOAP and WSDL are the major technologies for Web services. SAWSDL extends WSDL with pointers to semantics that are crucial for achieving automation. It does not specify a language for representing semantic models instead it provides mechanisms by which concepts from these models defined in the WSDL document can be referenced from within WSDL components as annotations. It facilitates to disambiguate the description of web services during automatic discovery and composition of web services. This defined concept requires being identifiable via URI references. It enables to annotate pure syntactic WSDL descriptions with pointers to semantic concepts in separate OWL file. Kopecky J (2007) discusses about interpretation of concepts and automation of service discovery, composition, selection, negotiation, mediation, and invocation.
1.8 SEMANTIC WIKI

A wiki is a Web-based system that enables collaborative editing of Web pages with the advantage of openness and flexibility. Wikis provide a Web-based text editor with a simple mark-up language to create content and to link pages as well as a versioning system to track content changes and full-text search for querying the wiki pages.

A semantic wikis discussed by Adam Souzis (2005), Baumeister (2008) supports metadata in the form of semantic annotations of the wiki pages and link relations between wiki pages. It offers semantically annotating links, wiki articles, and semantic search for querying, automatic or semi-automatic extraction of metadata from wiki articles to simplify the annotation process.

1.9 SEMANTIC WEB SERVICES

Web Services are self-contained, self-describing, and modular applications that can be published, located, and invoked across the Web. It can manipulate simple requests to complicated business processes. Web services can be deployed which can be discovered and invoked that runs on various applications and platforms. Web Services Description Language (WSDL) is an XML-based language that is used for describing the functionality offered by a Web service. A WSDL description provides machine-readable description of how the service can be called. Universal Description, Discovery and Integration (UDDI) are platform-independent, XML based registry to register, list and locate web services. Simple Object Access Protocol (SOAP) is a protocol specification for exchanging structured information in the implementation of web services. Semantic Web services were discussed by Cardoso (2005), Carlos P (2010) and Karthick et al (2003). The semantics included in the web services facilitates automatic discovery
and composition is also discussed by Traverso and Pistore (2004). It helps in application like collaborative learning as stated by Benlamri and Scott (2010).

Web services developed by service providers are syntactically described by WSDL document. Web services should be annotated with semantic descriptions for automation and enhancing discovery, composition and matching services. Semantic web service is discussed by Jorge Cardodo (2005). Annotation can be done using any ontology model by the service providers. SAWSDL refers the semantic ontology models in WSDL document by means of annotating those references. Web services should be registered in UDDI registry by using the semantically annotated WSDL document.

1.9.1 Quality of Service Preferences

Web services can be semantically annotated using SAWSDL to add semantics to web service concepts and enhances the discovery of web services. Generally the web services are discovered using function matching and I/O parameters matching. Since the web services available in Internet are vast in numbers, just the discovery of web services is not enough to ensure the Quality of Service (QoS) preferences of the clients requesting for web services. Instead users need for various levels of QoS is to be considered.

The QoS parameters for all the registered web services are calculated and stored in the database which is defined in the W3C working group-QoS (2003). While discovering web services, the QoS parameters are to be considered to satisfy the users. Various QoS parameters to be taken under consideration are Cost (Expensiveness), Response Time, Throughput, Availability and Reliability. The QoS parameters except cost have to be determined by service provider by testing process. Determined QoS values will be submitted to the system by service providers while registering their
web services and it should be accurate as it effectively influences the Grade Points of web services to be evaluated.

1.10 SEMANTIC WEB LANGUAGES

Semantic web has various languages like RDF, RDF(S), OIL, and OWL are developed to include more expressiveness. RDF/RDF(S) is not sufficient as ontology language, since more expressive languages for semantic web is needed.

1.10.1 Ontology Interchange Language (OIL)

OIL provides modeling primitives used in frame-based and Description Logic oriented Ontologies. OIL has formal semantics, reasoning support, modeling primitives and automatic consistency checking. The main contribution of OIL is that it is able to describe structured vocabulary with well-defined semantics.

1.10.2 DAML+OIL

DARPA agent markup language (DAML) + OIL is a semantic markup language for Web resources. It is an ontology language that describes classes and properties, and the set of axioms that assert characteristics of these classes and properties. It has sufficient expressiveness. The meaning is defined by standard model-theoretic semantics based on interpretations and these interpretations consist of domain of discourse and an interpretation function.

1.10.3 Web Ontology Language (OWL)

OWL is a standardized markup language for publishing and sharing data using ontologies on the Internet. A vocabulary extension of the Resource
Description Framework (RDF), OWL represents the meanings of terms in vocabularies and the relationships between those terms in a way that is suitable for processing. It is a language for defining and instantiating Web Ontologies. OWL ontology may include descriptions of classes, properties and their instances. The various types of OWL are OWL Lite, OWL DL and OWL Full. OWL Lite supports classification hierarchy and simple constraint features. OWL DL supports maximum expressiveness. OWL DL includes all OWL language constructs with restrictions such as type separation. OWL Full has maximum expressiveness and is syntax free.

1.11 SEMANTIC WEB TOOLS

1.11.1 Swoogle

Swoogle retrieves Semantic Web Documents (SWDs) that are in RDF and OWL or other formats. It provides links that has RDF content as response. This format is understandable by the machines as discussed by Tim Finin et al (2004). It extracts metadata for each discovered documents, and computes relations between them. It also collects metadata i.e., inter-document relations about the semantic web and reveals interesting structural properties of how resources are connected in semantic web.

1.11.2 Racerpro

RacerPro stands for Renamed ABox and Concept Expression Reasoner Professional. Description logics provide the foundation of international approaches to standardize ontology languages in the context of semantic web. It is used to manage semantic web ontologies based on OWL and reasoning engine for ontology editors. It is a repository with optimized retrieval engine because it is able to handle large set of data descriptions. It implements tableau calculus for very expressive description logic. It offers
reasoning services for multiple T-boxes and A-boxes. The services provided by RacerPro for OWL ontologies and RDF are tabulated in Table 1.1.

Table 1.1 Features of RacerPro

<table>
<thead>
<tr>
<th>Sl. No.</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Imports multiple resources through HTTP client</td>
</tr>
<tr>
<td>2</td>
<td>Finds implicit subclass relationship</td>
</tr>
<tr>
<td>3</td>
<td>Finds synonyms for resources</td>
</tr>
<tr>
<td>4</td>
<td>OWL-QL query to retrieve OWL instances and interrelationships</td>
</tr>
<tr>
<td>5</td>
<td>Incremental query answering</td>
</tr>
<tr>
<td>6</td>
<td>Checks OWL consistency and set of data descriptions</td>
</tr>
</tbody>
</table>

RacerPro has algebraic reasoning and provides min/max restrictions over the integers, linear polynomial (in-)equations over the real or cardinals with order relations, equalities and inequalities of strings. It does not support feature chains due to decidability issues.

1.11.3 JENA

Jena is an open source java framework for building semantic web applications developed by HP labs. It has a rule-based inference engine and provides a programmatic environment for RDF, RDFS, OWL and SPARQL. Its framework includes RDF application programming interface (API), reading and writing RDF in RDF/XML, N3 and N-Triples, OWL API, In-memory and persistent storage and SPARQL query engine. It also includes a document manager that assists with process of managing imported ontology documents.

The RDF API provides statement centric and resource centric methods for manipulating an RDF model as a set of RDF triples and set of
resources with properties respectively. It also has cascading method calls, built in support for RDF containers, integrated parsers and writers for RDF/XML (ARP), N3 and N-TRIPLES.

1.11.4 Protégé

Protégé is a free, open-source platform to construct domain models and knowledge-based applications with ontologies. Ontologies are important for many applications like scientific knowledge portals, information management and integration systems, electronic commerce and web services. Ontologies range from taxonomies, classifications, database schemas to fully axiomatized theories. They can be modeled using Frame-base or OWL. OWL organizes service description into four conceptual areas. They are process model, profile, grounding and service.

A process model describes how a service performs its tasks. It includes information about inputs, outputs, preconditions and results. It differentiates between atomic, simple and composite processes. A profile provides a general description of web service that is to be published and shared to facilitate service discovery. Profiles can include both functional properties (inputs, outputs, preconditions, and results) and nonfunctional properties (service name, text description, contact information, service category, and additional service parameters).

Grounding specifies how a service is invoked, by detailing how the atomic processes in a service’s process model map onto a concrete messaging protocol. A Service binds other parts together into a unit. The different parts of a service can be reused and connected in various ways.
1.12 UNIFIED MODELING LANGUAGE

The Unified Modelling Language (UML) is a standard for designing and documenting the systems. It is used in information system and object-oriented developments. It is widely used to create and view the application visually. It is used to visualize, modify, specify, construct and document the software projects. The features of UML Modeling have notations with elements that strongly define meaning. Hence it is scalable.

Class Diagram provides an overview of the target system by describing the objects, classes inside the system and the relationships between them as discussed by Robert Martin (1997). The UML class diagram can be used for modeling purpose since it depicts the overall things in an easiest way.

1.12.1 Class Elements

Class elements are the class having attributes (member variables), operations (member functions) and relationships with other classes. Figure 1.3 shows the notation of UML class.

<table>
<thead>
<tr>
<th>Class (Class name)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access specifier :</td>
</tr>
<tr>
<td>Attribute</td>
</tr>
<tr>
<td>Access specifier :</td>
</tr>
<tr>
<td>operations</td>
</tr>
</tbody>
</table>

*Figure 1.3 Notation for Class Diagram*

1.12.2 Multiplicity

Multiplicity indicates how many objects of the class are related to one object of another class. Different multiplicity is listed in Table 1.2. It includes upper bound and lower bound.
### Table 1.2 Indicators for Multiplicity

<table>
<thead>
<tr>
<th>Indicators</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0..1</td>
<td>Zero or one</td>
</tr>
<tr>
<td>1</td>
<td>One only</td>
</tr>
<tr>
<td>0..*</td>
<td>Zero or more</td>
</tr>
<tr>
<td>1..*</td>
<td>One or more</td>
</tr>
<tr>
<td>N</td>
<td>Only n (where n&gt;1)</td>
</tr>
<tr>
<td>0..n</td>
<td>Zero to n(where n&gt;1)</td>
</tr>
<tr>
<td>1..n</td>
<td>One to n (where n&gt;1)</td>
</tr>
</tbody>
</table>

#### 1.12.3 Association

An association represents a family of links. A Link is normally a line connecting two classes. An association can be named and the ends of an association can be adorned with role names, ownership indicators, multiplicity, visibility, and other properties. There are four different types of association. They are bi-directional, uni-directional, Aggregation and Reflexive. Bi-directional and uni-directional associations are the most common ones. The association is between two or more class that specifies connections among their instances.

#### 1.12.4 Aggregation

A binary association representing part-whole relationship is Aggregation. It is a weak form of Association. As aggregation is a type of association, it can also be named and have the same adornments as that of association. It is shown with hollow diamond at the aggregate end of the association line. If one class is said to be composed of another class then it is represented by using composition relationship, but the contained classes do
not have a strong life cycle dependency on the container. The black diamond symbol represents composition relation. For example each instance of class word is seems to contain the instance of the class character.

1.12.5 Generalization

It is a binary directed relationship between a super class also called as base class and a subclass also called as derived class. So generalization is also called "Is A" relationship. A generalization is a line with an arrowhead pointing to the super class.

1.13 MAPPING UML CLASS DIAGRAM TO OWL

UML models are created using several tools but the main drawback is that it lacks semantics. UML class diagram can be translated into XML Metadata Interchange (XMI) format which is treated as a web resource. UML class diagram and ontology share similar features which makes it possible to map the class diagram to OWL. The UML model has inconsistency and redundancy in its diagram which will be ported to the OWL code when it is mapped. So, it is essential to identify these inconsistencies in a formal way. Several works has been done in mapping UML to OWL using first order logics and description logics.

Robert Martin (1997) discusses Ontology, which is the building block of semantic web, is created during the modeling stage by converting UML models to OWL. Formalization of ontologies from UML model is done by using First order logic and Description logics. Initially the UML diagrams can be formalized by means of First order logic, then formalization is encoded into fragment of Description Logic.
Luis Iribarne and Nicolas (2011) talk about Description Logic (DL) obtained is converted into OWL ontology. The steps involved are i) generate the UML class diagram, ii) convert the class diagram to first order logic (FOL) using UML to FOL representation iii) Represent the FOL in Description Logic and iv) MapDL to OWL. Domain specific semantics is added using First order logic but description logic supports completeness, give expressiveness and reduce inconsistency in the UML diagrams in a formal way. Also separate reasoning of OWL is not required.

1.13.1 Similarities between UML and OWL

The common properties between UML and OWL are discussed by Moein Mehrolhassani (2008) which is shown in Table 1.3. These properties aids to map the UML class notions to OWL syntaxes easily. These concepts, properties and relationship can be mapped to OWL that can be reasoned. The UML model has inconsistency and redundancy which will also be reflected when mapped and when it is reasoned will produce incorrect results. This is to be identified and removed.

Table 1.3 Similarities of UML and OWL

<table>
<thead>
<tr>
<th>UML</th>
<th>OWL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class</td>
<td>Class</td>
</tr>
<tr>
<td>Instance</td>
<td>Instance</td>
</tr>
<tr>
<td>Attribute, Binary Association</td>
<td>Property</td>
</tr>
<tr>
<td>Sub class</td>
<td>Sub class</td>
</tr>
<tr>
<td>N-ary Association, Association class</td>
<td>Class, Property</td>
</tr>
<tr>
<td>Enumeration</td>
<td>One of</td>
</tr>
<tr>
<td>Navigable, Non-navigable</td>
<td>Domain, Range</td>
</tr>
<tr>
<td>Multiplicity</td>
<td>Min cardinality, Max cardinality, Inverse of</td>
</tr>
<tr>
<td>Package</td>
<td>Ontology</td>
</tr>
</tbody>
</table>
1.13.2 UML Class Diagram to Description Logic Conversion Rules

UML class diagram consists of Class elements and the relationship between the classes. The class consists of attribute, operations and relationships with other Classes. The UML class diagram is to be consistent in order for it to be reasoned correctly. So to check the consistency of the UML model, description logics is used. Satisfiability of the UML class diagram is also discussed by Alessandro et al (2010). Description logic for data modeling is conferred by Diego Calvanse et al (2003).

An UML class is represented in description logic by an atomic concept C and each attribute of type T for Class C is represented by an atomic role ‘a’ like C: C ⊑ ∀a.T. Multiplicity, Association and aggregation can also be represented in DL. The multiplicity [i::j] of attribute ‘a’ for a Class ‘c’ is represented as C ⊑ (≥ i a) ∩ (≤ j a) and when multiplicity[j..*] of attribute ‘a’ for a Class ‘c’ is represented as C ⊑ [j...*]. When the multiplicity is [0;*] the whole assertion can be omitted. If the multiplicity is missing or not defined, the default multiplicity is [1::1] taken and its assertion becomes C ⊑ a ∩ (≤ 1 a).

In UML class, association is represented in two ways. They are association with class and association without class. Figure 1.4 shows the DL representation for the Associations.

<table>
<thead>
<tr>
<th>Name</th>
<th>Notation</th>
<th>Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Without Association Class</td>
<td><img src="image" alt="Diagram" /></td>
<td>$C1 \cong \forall A.C2 \cap \forall A.C1$</td>
</tr>
<tr>
<td>With Association Class</td>
<td><img src="image" alt="Diagram" /></td>
<td>$C1 \cong \geq \text{min} r1(A) \cap \leq \text{max} r1(A)$</td>
</tr>
</tbody>
</table>

Figure 1.4 DL representation for Association
In Multiplicity each instance of C1 is connected through A to at least $\min_1$ and at most $\max_1$ instances of

$$C2: C1 \subseteq (\geq \min_1 A) \cap (\leq \max_1 A)$$

and each instance of C2 is connected through A to at least $\min_2$ and at most $\max_2$ instances of

$$C1: C2 \subseteq (\geq \min_2 A^-) \cap (\leq \max_2 A^-)$$

Figure 1.5 shows the notation for generalization. This generalization can be represented as

$$C \subseteq C_1 \sqcup \ldots \sqcup C_k$$

and it also holds disjointness between the sub classes as

$$C_i \subseteq \neg C_j \text{ for } 1 \leq i < j \leq k$$

Figure 1.5 Notations for Generalization

Figure 1.6 shows the DL representation for Aggregation. Aggregation is a type of association. Aggregation between two classes can have the association line with and multiplicity at its terminals, in its association line along with its Description Logic Statements.
Consistency of the UML class diagram can be checked using several reasoning properties. Frequent inconsistencies among classes, attributes, and associations in UML class diagrams that affect the OWL are Duplicate name errors, attribute value type errors, generalization and Disjointness, multiple inheritances also stated by Ken kaneiwa and Ken Satoh (2005). Duplicate name error occurs when two of classes, attributes and association are with same name. Attribute value type errors arises if two classes with same attribute, but with different type namely a: T1 and a: T2 where ‘a’ is attribute, T1 and T2 are data types or classes. David Bell et al (2007) talks about the approaches to handle inconsistency in the DL based Ontologies.

Generalization and disjointness, appears when a class C has a super class C_k but the classes C and C_k are defined as disjoint to each other in the constraint of a class hierarchy which is represented in Figure1.7. Multiple inheritances inconsistency occurs when there is conflict with the association and super association.
Figure 1.7 Generalization and Disjointness Inconsistency

The above discussed inconsistency is said to be there and detected using DL when at least one of the reasoning properties is not satisfied to the obtained set of Description logic statements. The reasoning properties like subsumption, satisfiability, equivalence and disjointness in description logics are applied to check the consistency of the model. The format for equivalence is

\[ \text{Equivalence (C, D) when subsumes(C, D) and subsumes (D, C)} \]

where C and D are Classes. Subsumption is the main property which detects the presence of cycle in UML model and helps to identify all the inconsistent statements. The format is

\[ \text{C is Subsumed to be D or D is Subsumed to be C \( \subseteq \), C \( \subseteq \) D} \]

where C and D are Classes. Disjointness checks if two classes denote different set. If they denote different set then put the attributes in different set. Once all concept inference services like Subsumption, Equivalence and Disjointness are satisfied then it leads to satisfiability and the domain is said to be consistent. The detected inconsistencies are to be resolved using different approaches. After all the inconsistencies are removed, the obtained consistent DL statements are mapped to OWL which will also be consistent.
1.13.4 Mapping Description Logic to OWL

OWL representation is vital in automate application development as it assists machines to understand the content.

Table 1.4 Mapping Rules for DL to OWL

<table>
<thead>
<tr>
<th>Description Logic</th>
<th>OWL</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1 \cap C_2 \cap \ldots \cap C_n$</td>
<td>IntersectionOf</td>
</tr>
<tr>
<td>$C_1 \cup C_2 \cup \ldots \cup C_n$</td>
<td>UnionOf</td>
</tr>
<tr>
<td>$\neg C$</td>
<td>ComplementOf</td>
</tr>
<tr>
<td>$X_1 \sqcup X_2 \ldots \sqcup X_n$</td>
<td>one of</td>
</tr>
<tr>
<td>$\forall P.C$</td>
<td>allValuesFrom</td>
</tr>
<tr>
<td>$\exists P.C$</td>
<td>someValuesFrom</td>
</tr>
<tr>
<td>$\leq nP$</td>
<td>maxCardinality</td>
</tr>
<tr>
<td>$\geq nP$</td>
<td>minCardinality</td>
</tr>
<tr>
<td>$C_1 \sqsupseteq C_2$</td>
<td>subClassOf</td>
</tr>
<tr>
<td>$C_1 \equiv C_2$</td>
<td>equivalentClass</td>
</tr>
<tr>
<td>$P_1 \sqsubseteq p_2$</td>
<td>subPropertyOf</td>
</tr>
<tr>
<td>$P_1 \sqsupset p_2$</td>
<td>Equivalent Property</td>
</tr>
<tr>
<td>$P_+ \sqsubseteq P$</td>
<td>transitive Property</td>
</tr>
<tr>
<td>A:C</td>
<td>Type</td>
</tr>
<tr>
<td>(A,b):R</td>
<td>Property</td>
</tr>
</tbody>
</table>

Hence, from the UML model, consistent Description Logic statements are generated and are mapped to the corresponding OWL statements based on the similar properties that exist between them. These properties are stored and retrieved for mapping. Every class elements and its
relationship can be transformed to the OWL code. Table 1.4 explains the mapping rules between Description Logic and OWL.

1.14 LITERATURE SURVEY

Sebastian et al. (2008) discusses semantic wikis and Mediawiki Manual (2010) gives an overview of the architecture of the Mediawiki. It has four layers user layer, network layer, logic layer and data layer. Logic layer is the core of Media wiki architecture and data layer has all the schema definition in it. The database layout of Mediawiki is normalized that adding a new table needs no alterations in any other table. The metadata about each page is stored in the table called page table and the actual content of the page is stored in a separate table called text table. So, semantic annotation of wiki pages can be done and the parser can be enhanced such that it identifies the annotated tags. The microdata annotation of wiki pages using semantic HTML elements and attributes reduces the user effort than the earlier proposal of semantic wiki using RDF such as Kawawiki and Rhizome.

Akerkar and Lingras (2008) highlighted the parsing the contents on a page. They state that search engine or any other software that needs to be extracted from pages need not search the entire web page instead it is enough to search the variable data i.e. the contents excluding the fixed (template) part. As Media wiki has a standard template for its web pages, it is easy to search the content of the wiki pages by the parser. So, parser needs to be augmented to enhance the search of wiki pages.

As stated earlier, Luis Iribarne and Nicolas (2011) investigated that Ontology is the building block of semantic web is created during the modeling stage by converting UML models to OWL. Formalization of ontologies from UML model is done by using First order logic (FOL) and Description logics. Initially the UML diagrams can be formalized using FOL
and encoded into fragment of Description Logic. Standard Description Logic obtained is converted into OWL ontology. FOL will give semantics for specific domain and the Description Logic will support completeness, gives expressiveness, reduce inconsistency in the UML diagram that helps in obtaining consistent OWL for that domain. So, OWL need not be reasoned separately.

Moein Mehrolhassani (2008) presents an approach to develop OWL from modeling. UML models can be understandable only to humans, to make it readable for machines it is converted to OWL. Transformation of class diagram to OWL is done because similarity exists between the two languages and many enterprises have their schema in UML. Straightforward and new approaches of conversion rules for each UML model is proposed for mapping UML to OWL.

Ontology Definition Metamodel (2008) gives an approach for converting UML to OWL using its mapping rules. UML class diagram has properties like class, attribute, package, binary associations, association class, generalization, multiplicity and enumerations. OWL has elements like class, properties, ontology, cardinality, range, domain and one of. The translation from UML to OWL is straight forward since both supports some common features like modules, associations, multiple inheritance and multiplicity. The module structures are called package in UML and ontology in OWL. Association that provides relationship to the class in UML is represented as Properties in OWL. Multiple inheritances in UML have subclass related to the super class that is represented as super class to be the union of the subclasses in OWL. UML class has a cardinality restriction which is represented by using mincardinality and maxcardinality depending upon its value. Because of these common features, UML can be directly mapped into OWL.
Daniela Berardi et al (2003) gives information about reasoning in UML Class diagram using description logic statements. Franz Baader et al (2006) discusses about how reasoning helps in detecting their inconsistencies and redundancies. Initially Class diagram is expressed in DL statements to make it easier for reasoning and is done with the help of DLR translations. DL helps with properties like Disjointness of classes and partitions of classes into subclasses from the class diagram. DLR assertions are converted to ALCQI assertions since the DL based reasoning system accept the variants of the latter only.

Ken Kaneiwa and Ken Satoh (2005) focused on consistency checking of the UML class diagrams. An algorithm is designed which identify the inconsistency triggers that cause logic inconsistency among the classes, attribute and association by recognizing the problems like combination of disjointness/completeness constraints, attribute multiplicities and multiple inheritance. They identify the inconsistencies in the class diagram by translating them to First order logic and gave expressiveness to the diagram by eliminating disjointness constraints and multiple inheritance.

Jens Lehmann and Lorenz Buhmann (2010) proposed an approach for debugging OWL ontology. If Debugged OWL ontology is inconsistent then it will calculate root and derived unsatisfiable classes. The erroneous axioms in the root class were modified to preserve the consistency which is also discussed by Heba et al (2011). Repairing a problem involves editing or deleting an axiom. In case of deletion, the particular axiom only removed but in case of editing axioms, newly added axioms may also leads to inconsistency. Deletion has minimal impact therefore preview of impact of axioms lost and retained will be shown to user. The users have to decide on deleting axioms in the list to remove inconsistency.
Aditya Kalyanpur et al (2006) proposed a method for resolving inconsistency by using ranking approach. In this approach, OWL for the particular domain is reasoned using pellet reasoner and inconsistent class is derived. For each inconsistent class, erroneous axioms are calculated and finally each erroneous axiom is ranked. Ranking is based on two parameters like Impact and Arity. The Arity is number of occurrence of particular axioms in the other inconsistent classes. Impact is number of axioms will get affected by removing the axioms from list and then ranking is done with the total of Arity and Impact values. Based on the ranking, the error axioms are removed and inconsistency is resolved.

Ian Horrocks et al (2005) proposed a concept of converting Description logic statements into OWL. DL is the knowledge representation language which is used to check the consistency in UML models. DL is mapped into OWL because OWL is the building block of semantic web since it is in machine understandable form. It provides syntax for direct mapping of class, attribute, cardinality, association, generalization and aggregation of description logic statements into its corresponding OWL statements. This paper also covers about RDFS, its syntax and semantics and provides information on OWL precedence like DAML, OIL. The semantic web does the annotations process using classes and relationship defined in ontologies for providing services.

Rama Akkiraju et al (2005) proposed that web service use existing semantic models like OWL, RDF with the help of WSDL documents. The interface WSDL document facilitates the web service by referring semantic models using its WSDL elements like types, message, ports, and bindings. The semantic information specified in this document is input, output, precondition and web service operation. For this process, initially the developer includes the semantics and operational level details for the
application model inside the WSDL documents. The web service developer then annotate their web services for the existing semantic models like OWL.

Ting-Xin Song et al (2008) proposed that SAWSDL is the World Wide Web Consortium's (W3C) initiative in standardizing the technologies for Semantic Web Services. The major technologies for Web services are SOAP and WSDL. SAWSDL extends WSDL with pointers to semantics that are crucial for achieving automation. SAWSDL itself does not provide any specific semantics of web services. It enables us to annotate the purely syntactic WSDL descriptions with pointers to semantic concepts in separate OWL file that defines the Ontology system. Software systems can interpret these concepts to (partially or fully) automate such tasks as service discovery, composition, selection, negotiation, mediation, and invocation. Web Services should be developed and its semantic descriptions should be annotated using SAWSDL. Ontology model should be created using Protégé and it can be analysed by Jena API. System should also have the transformation files to transform the semantic concepts which are in OWL to XML schema and vice versa. SAWSDL is a simple extension of WSDL using the extensibility elements. It has two basic types of annotations, the model reference and the schema mapping.

Elena Simperl (2009) discusses about importance of ontology and how to reuse these Ontologies. One of the ways is to translate the XML data to OWL which facilitates uniform access of libraries and repositories. They translated UMLS to OWL and created taxonomy of semantic types as classes and a taxonomy of relations as properties for Metathesarus. They also performed consistency checking on these Ontologies and found few differences and difficulty to reason the ontology. Also, interoperability increases if existing resources are reused.
Valentina Presutti and Aldo Gangemi (2006) discusses about the importance of identification of web resources. They proposed a formal model for the categorization of the entities like documents, metadata, abstract concepts, physical things and events. The elements are defined as resource, web resource, proxy resource, semantic resource and are expressed in IRE pattern (Identifiers, Resources and Entities which helps in referencing and accessing web resources.

Kopecky et al (2007) described the Web Services Description Language (WSDL) specifies a way to describe the abstract functionalities of a service and concretely how and where to invoke it. The WSDL 2.0 W3C Recommendation does not include semantics in the description of Web services. So, two services can have similar descriptions but may have different descriptions. SAWSDL provides mechanism to include semantic annotation in the WSDL components. No additional container element defined instead identifier is referenced. SAWSDL provides two constructs: model reference and schema mapping to annotate XML schema documents. The model reference provides a generic link from an XML structure to a semantic model. It assists to determine if it matches the client requirements. The schema mapping is used to check the structural mismatches between semantic model, service input and output after the discovery of web services. XML Schema components such as simple types (xs:simpleType), complex types (xs:complexType), elements (xs:element) and attributes (xs:attribute) can be annotated using modelReference. The (wsdl:fault) element describe about the fault that is could be associated with the operation and does not give any error message. The (wsdl:interface), (wsdl:operation) is used to annotate, categorize, describe operation and specify behavioural aspects.

Yousefipour et al (2010) proposed that Discovery of web services can be divided into two sub processes namely functional matching and I/O
parameter matching. Functional matching ensures that the functional requirements of user will be satisfied by discovered web services and I/O parameter matching ensures the same for I/O requirements. Using the SAWSDL, we can annotate the functional and I/O descriptions of web services in its WSDL documents. Such annotations will ease the discovery of web services which matches the functional and I/O requirements of client’s request. Due to function semantics is a kind of ontology concept which is suited with the function in service request and description, so degree of function matching can be calculated according to matching degree of two ontology concepts. Function matching includes category (C) and description (D) matching and its matching algorithm is defined. Web services should provide all the output parameters required by client and it should be able to operate with the set of input parameters provided by client. Such web services meant to satisfy input and output constraints of client’s request and those should be shortlisted for further processing.

W3C Working Group Note (2003) described the web services usage are increasing and the quality-of-services (QoS) is becoming very important for both service providers and service requestors. Due to the dynamic and unpredictable characteristics of the web services, it is a difficult task to provide the desired QoS for web service users. The performance of a web service represents how fast a service request can be completed. It can be measured in terms of response time, throughput, availability, reliability, latency, execution time and transaction time. Response time is the time required to complete a web service request. Throughput is the number of web service requests served in a given time interval. The availability is the probability that the system is up and related to reliability. Reliability is the ability of a web service to perform its required functions under stated conditions for a specified time interval. The reliability is the overall measure of a web service to maintain its service quality. Latency is the round-trip
delay (RTD) between sending a request and receiving the response. Execution time is the time taken by a web service to process its sequence of activities. Transaction time represents the time taken by the web service to finish one complete transaction. The web service should provide higher throughput, high reliability, faster response time, lower latency, lower execution time, and faster transaction time. Cost (Expensiveness) of the web services should be optimal that is determined by the service providers and it should be satisfied by service requesters.

1.15 EVALUATION OF THE STUDY

The survey for this thesis exemplifies that millions of web resources like web pages, wiki pages and web services are available in the World Wide Web which is also stated by Albert et al (1999). Semantic web the extension of WWW tries to add meaning to the resources to enable machines to understand the content of the web resources. All these resources cannot be substituted with new meaningful resources. Machines can interpret these resources if they are added with some additional information called annotation or it can be referred to an external resource to set some relationship between resources. Ontology has a significant role in semantic web. Ontology can be designed and developed using web ontology language. W3C semantic web recommendations developed standard format and languages to add semantics or to refer additional resources or URI. Some of them are Resource description framework (RDF), HTML5 microdata, Web Ontology Language (OWL), Semantic annotation for web services description language (SAWSDL). Also, the study reveals that in the modeling phase itself OWL code can be generated and stored it in a repository that can be accessed by other domain specific applications.
Hence the study substantiates that

1. The resources need to be represented in different format to increase machine understandability and to enhance search.

2. There is a need to include additional information in the web resources or annotation is to be done.

3. In modeling phase, OWL code is generated from UML class diagram and inconsistency is detected. But there is a need to resolve the detected inconsistencies. Formal methods need to be applied and resolved.

4. Resources like web services need to be annotated to make it accessible automatically by machines and SAWSDL can be used to represent web services.

5. Also, the discovery of web services based on non-functional requirements need to be considered and also the preferences of the user is to be satisfied.

1.16 CONTRIBUTIONS

This thesis makes contributions towards better representation of web resources and enabling search of these web resources. They are as follows:

a. The main objective of this work is to propose a web resource transformation architecture in semantic web that enables transformation of various web resources like web pages, wiki pages, web services and modeling these resources and to create a knowledge repository that can be reused by other applications.
b. This thesis presents conversion of webpages to RDF format and generating content summary for a user query.

c. This thesis intent to annotate wiki pages using HTML5 microdata to include semantics thereby enhancing the search of these wiki pages.

d. At modelling level, the system intents to generate consistent OWL code from class diagram and propose an algorithm to detect and resolve inconsistencies by analyzing the dependency.

e. This thesis presents semantic annotation of web services at server level and suggesting the web services by considering the Quality of services (QoS) of the web services. It also analyse the preferences of the user and usage of Web services.

1.17 ORGANISATION OF THESIS

The rest of the thesis is organized as follows: Chapter 2 is organized to discuss about the proposed Web Resources Transformation Architecture and gives an overview of the system. The Chapter 3 discusses about the representation of Web pages in RDF, the system design and the working of HTML generator. It also discuss about the results obtained.

Chapter 4 discusses about Wiki page representation and semantic search of these pages. It presents the modified design of the MediaWiki system, the annotation done to enhance the search and talks about flow of the search and microdata annotation

Chapter 5 presents the UML to owl conversion and detection of inconsistency using formal methods. The proposed algorithm detects the various inconsistencies and resolves the multiple inheritance inconsistency.
Chapter 6 describes the modified algorithm which considers the dependency and compares with the existing approach. Also the results obtained for various domains are discussed.

Chapter 7 and 8 describe about the semantic annotation of web services and how it helps in discovering the services. Also the recommendation of the important web services as per the user preference by obtaining the QoS is discussed. It compares the user request for different QoS. Finally, the thesis concludes and presents suggestions for future work in Chapter 9.