B.Sc. in Computer Science and Engineering Thesis

Entry Creation for An Educational Institution and Searching in The Semantic Web

Submitted by

Syeda Nyma Ferdous
201014054

Jannatut Tabassum
201014059

Abdullah Al Noman
201014032

Supervised by

Dr. Muhammad Masroor Ali
Professor
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

.

Department of Computer Science and Engineering
Military Institute of Science and Technology

CERTIFICATION

This thesis paper titled **"Entry Creation for An Educational Institution and Searching in The Semantic Web"**, submitted by the group as mentioned below has been accepted as satisfactory in partial fulfillment of the requirements for the degree B.Sc. in Computer Science and Engineering on December 2013.

Group Members:

Syeda Nyma Ferdous
Jannatut Tabassum
Abdullah Al Noman

Supervisor:

_____

Dr. Muhammad Masroor Ali
Professor
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

CANDIDATES' DECLARATION

This is to certify that the work presented in this thesis paper is the outcome of the investigation and research carried out by the following students under the supervision of Dr. Muhammad Masroor Ali, Professor, Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka, Bangladesh.

It is also declared that neither this thesis paper nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

_____

Syeda Nyma Ferdous
201014054

_____

Jannatut Tabassum
201014059

_____

Abdullah Al Noman
201014032

ACKNOWLEDGEMENT

We are thankful to Almighty Allah for his blessings for the successful completion of our thesis. Our heartiest gratitude, profound indebtedness and deep respect go to our supervisor Dr. Muhammad Masroor Ali, Professor, Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, for his constant supervision, affectionate guidance and great encouragement and motivation. His keen interest on the topic and valuable advices throughout the study was of great help in completing thesis.

We are especially grateful to the Department of Computer Science and Engineering (CSE) of Military Institute of Science and Technology (MIST) for providing their all out support during the thesis work.

Finally, we would like to thank our families and our course mates for their appreciable assistance, patience and suggestions during the course of our thesis.

Dhaka
December 2013

Syeda Nyma Ferdous
Jannatut Tabassum
Abdullah Al Noman

ABSTRACT

The main purpose of the Semantic Web is driving the evolution of the current Web by enabling users to find, share, and combine information more easily. Almost all of our activities on the web are limited to search, integration and data mining. In the current web it is both tedious and time consuming to find any information that we are looking for. Besides it doesn't have any ability to infer anything from available data and make intelligent decision for us. But with the development of semantic web it gets much more efficient and faster to obtain only the required information by ignoring all other unnecessary data. Moreover it understands the meaning of data and can make intelligent decision for us based on the available data to automate the process. But for this automation to take place, first we need to markup the web pages, add some extra information to it, hence create some RDF documents. For creating these RDF documents there are available many tools and software. After creation, these documents need to be validated to make sure that they are indeed error free. Then these documents can be deployed on the web.For the search engine to be able to retrieve them, URL of the container websites must be submitted to it. After a certain period of time search engine can index the documents and the required items can be retrieved from the web when searched. Here we generate RDF for an educational institution, deploy them on the web and search them with a semantic web search engine. These documents in turn make it possible for the semantic web to retrieve the necessary information more efficiently and speed up the automation process.

TABLE OF CONTENT

References

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION

RDF : Resource Description Framework

SWD : Sematic Web Document

SWO : Semantic Web Ontology

SWDB: Semantic Web Database

XML : Extensible Markup Language

CHAPTER 1

INTRODUCTION

1.1 General Discussion

Let us think of a world wide web which is not totally human-processed, a web of data where people can find the documents that they need only and no other unnecessary documents. This can be possible only if the device understands what people really want. Few years ago, it seemed like a dream, but now they are on the way to reach this dream. Few years ago when someone said, "I found this on the web", people knew he or she was talking about using a web search engine. But the scenario has changed now, at least a little. World Wide Web has evolved since then. Today along with that traditional web, people talk about semantic web, a web whose resources are not only for human-use but also machine-readable. The current web is made up of so many documents that machine doesn't understand its meanings and can't make efficient decision during search. Semantic web holds the idea of presenting web documents in such a way that it is understood by the machines. In traditional web basically three major activities are performed. Search, integration, web data mining [**?**].

1.1.1 Search

This is probably the most common uses of internet. This actually means locating or accessing information on the web. But it gets quite frustrating often. For example, to search a word 'web' using a common search engine,almost a million listings will be found. Some of them may include web of spiders, while actually information of the web of internet was being searched. And to find out required information regarding web, user might have to go through a number of listings, which is tedious as well as a time consuming process. Now the task of searching would be much more efficient if one could get only the required information. But the traditional web doesn't actually work that way. It maintains an index table by crawling all the web pages and matching the words. So, it serves all web pages which

match the contents and according to ranking of pages.

1.1.2 Integration

Integration means combining and aggregating resources on the web so that they can be collectively useful. Often there are a number of steps of searching involved for actual search item. For example, if people need to buy an item of their interest, first they search based on the category of items that meet their budget and then they may search for the closest market place available from where they may buy that item. So, generally there are multiple steps of searching involved to meet the final goal. So, it would be very much convenient if there would be an automatic agent working, that could perform all the necessary steps and serve users with the final result. This process is known as integration. But with the traditional web there is no way that users could get that advantage. With the help of semantic web it could be possible to create the automatic agent based on different applications.

1.1.3 Web Data Mining

The Internet can be viewed as a huge distributed database, so web data mining refers to the activity of getting useful information from the Internet. For example, there is no way for the crawler to understand the meaning of the information served in a website. So, it can't take decision based on the data served, and doesn't have any knowledge of the data or content of the webpage, so that it would have been able to extract the actual information that is required by the user. But it would have been great if users had an automatic agent perform for them, able to understand the meaning of data or content in the webpages, compare based on that information and come up with the required result.

1.2 Semantic Web

Semantic web, the meaningful web is the movement to reach the goal of more easily processed web than the current web by the World Wide Web Consortium. The word semantic stands for 'meaning', 'the study of the significant meaning'. A question can arise in mind that the traditional web is understandable to the human, so to whom the semantic web is

going to make its resources meaningful. The answer is- to the machines. So, semantic web is the extension of current web which allows the machine to understand the data on the web. It is done adding metadata in the documents. By encouraging the inclusion of semantic content in web pages, the Semantic Web aims at converting the current web into a "web of meaningful data". The main purpose of the Semantic Web is driving the evolution of the current Web by enabling users to find, share, and combine information more easily.

Tim Berners-Lee originally expressed the vision of the Semantic Web as follows [**?**]: "I have a dream for the Web [in which computers] become capable of analyzing all the data on the Web- the content, links, and transactions between people and computers. A 'Semantic Web' , which makes this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines."

1.2.1 Semantic Web Development

The discussion above is about what semantic web is, and then comes how the semantic web is developed. A dedicated team of people at theWorld Wide Web Consortium (W3C) is working to improve, extend and standardize the system. So many languages, publications, tools have already been developed. The basic task of searching, integration and web data mining is performed more efficiently in semantic web. To improve searching web pages are associated with metadata elements. Metadata can be defined as data about data. That is data that describe the actual content of the webpage. When crawler crawls through the web pages it extracts the metadata elements and take necessary actions based on those data. This is how searching is speeded up. Depending on the metadata elements it understands the actual information or content of the web pages and able to take decision for the users.

The prerequisite for the development of the cherished semantic web are the terms "semantics" , "metadata" and "ontologies" . In particular, these terms are used as everyday terminology by researchers and practitioners, spanning a vast landscape of different fields, technologies, concepts and application areasfor the development of sematic web ontology. For the web, ontology is the exact description of web information and relationship among them. To create and implement this ontology for the semantic web, the basic building block is RDF. OWL is also a part of the semantic web vision. Both RDF and OWL are written

in XML. XML is the basic syntax for them and RDF holds the semantics. XML Schema, XML Namespace, RDF Schema these are associated with the semantic web. A web definitely needs a search engine. As a consequence a search engine for the semantic web is developed, named "SWOOGLE" . SWOOGLE crawls for the metadata index them and make the search operation easy for the users.

1.3 Document Creation and Manipulation

The work for this thesis was to create entry for an educational institution and searching in the semantic web. RDF documents for an educational institution have been created using RDF creation tool. Then the documents have been deployed in the web and URL is submitted in the search engine for indexation. After the indexation has been completed, documents can be retrieved by searching in swoogle.

1.4 Comparison and Quality Assesment Issues

After creating the RDF documents it is necessary to verify whether these documents are syntactically correct. It is also needed to check whether these documents fulfill the requirement. That means whether the search engine is able to locate the documents and retrieve necessary information from them.

CHAPTER 2

SEMANTIC WEB WORLD

2.1 Semantic Web Search Engine

At the user end of an web, search engine is a most important thing. A web document can not be found if there is no search engine. As users are cherishing for a machine-understandable web, so there is also a need of a search engine that can understand what users actually wants. As a result of a hardwork by the developers, a search engine for the semantic web has been invented. "Swoogle" is the search engine for semantic web. For swoogle, semantic web is the web of semantic web documents(SWD). Swoogle considers files with extension .rdf, .owl or .rss as SWD.

2.1.1 Swoogles Architecture

Swoogle's architecture can be broken into four major components: SWD discovery, meta-datacreation, data analysis and interface. This architecture is data centric and extensible; different components work on different tasks independently [**?**].

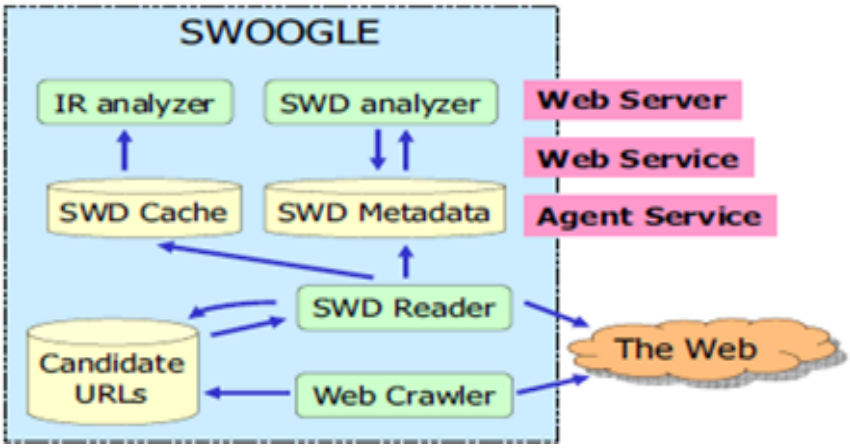


Figure 2.1: The Architecture of Swoogle [2]

Here comes the basic architectural description of Swoogle. The SWD discovery component

discovers the potential SWDs throughout the Web and keep up-to-date information about SWDs. The metadata creation component generates objective metadata about SWDs by capturing the snapshot of a SWD in both syntax and semantic level. The data analysis component uses the cached SWDs and the created metadata to derive analytical reports, such as classification of SWO and SWDB, rank of SWDs, and their index of SWDs. The interface component provides data service to the Semantic Web community. Swoogle has a crawler which extracts the metadata for each document and index the data into information retrieval for next search.

2.1.2 Uses of Swoogle

Finding appropriate ontologies:

Typically, an RDF editor allows a user to load an ontology, which user can use to make assertions. But finding the right ontology to load is a problem, and the lack of an adequate solution has led to ontology proliferation. A user can query Swoogle for ontologies that contain specified terms anywhere in the document (including comments). They can also search for ontologies that contain specified terms as Classes or Properties; or for ontologies that are about a specified term (as determined by our IR engine). The ontologies returned are ranked according to the Ontology Rank algorithm, which finds at which extent the ontologies are being used. This use of Swoogle will both ease the burden of marking up data and contribute to the emergence of canonical ontologies.

Finding instance data:

The semantic web enables the integration of distributed information. But first, the information must be found. A Swoogle user can query for all instance data about a specified class, or on a specified subject. The triples of the returned SWDs can then be loaded into a knowledge base for further querying.

Studying the structure of the semantic web:

The meta-data computed by Swoogle will provide structural information about the semantic web. How the data is connected, which documents refer to an ontology, which ontologies does a document refer to, what relationships (importing, using terms etc.) exist between two documents, where the graph is of most density, etc.

2.1.3 Current Status and Future Work

Swoogle is an ongoing project. It is undergoing constant development. A general user can query with keywords, and the SWDs that matches those keywords are returned in ranked order. The ranking algorithm ranks Semantic Web Ontologies(SWO) higher than Semantic web DataBase(SWDB)s; thus, Semantic web ontologies using those query terms will be returned before SWDBs using those terms. The highest ranked SWDs typically are the base ontologies that define the semantic web languages, such as the RDF or OWL definitions, which all SWDs must import.

For advanced users, an advanced search interface is provided which essentially allows them to fill in the constraints to a general SQL query on the underlying database. The user can query using keywords, content based constraints (type of SWD, number of classes/properties/ individuals), language and encoding based constraints (N3 vs XML), and/or the Rank of the document. At present, the metadata are stored in a mySQL database, and indexes about 11000 SWDs [**?**].

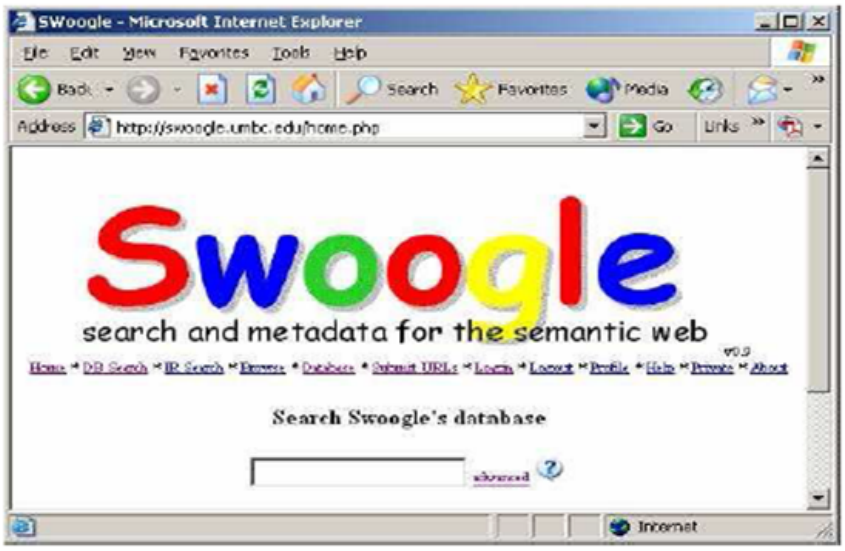


Figure 2.2: Swoogle Interface

2.2 Ontology

Ontologies are considered one of the pillars of the Semantic Web, although they do not have a universally accepted definition. A (Semantic Web) vocabulary can be considered as a special form of ontology, or sometimes also merely as a collection of URIs with an

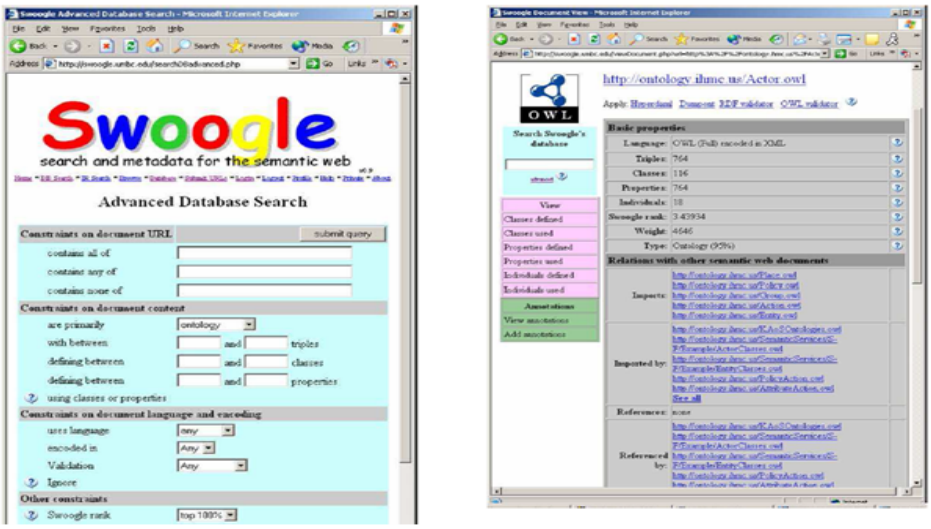


Figure 2.3: Swoogle Query Result [2]

described meaning. Ontologies are usually assumed to be accompanied by some document in a formal ontology language, though some ontologies do not use standardized formats for that purpose.

2.2.1 Basics of Ontology

An ontology is a formal specification of a shared conceptualization. The main thread of ontology in the philosophical sense is the study of entities and their relations. Ontology asks the questions: What kinds of things exist or can exist in the world, and what manner of relations can those things have to each other. It is less concerned with what is than with what is possible.

Several aspects of defining ontologies are :

First, this definition states that ontology is used to describe and represent an area of knowledge. In other words, ontology is domain specific [**?**]; it is not there to represent all knowledge, but an area of knowledge. A domain is simply a specific subject area or sphere of knowledge, such as photography, medicine, real estate, education, etc.**Second**, ontology contains terms and the relationships among these terms. Terms are often called classes, or concepts; these words are interchangeable. The relationships between these classes can be expressed by using a hierarchical structure: superclasses represent higher-level concepts and subclasses represent finer concepts, and the finer concepts have all the attributes and features that the higher concepts have.**Third**, besides the mentioned relationships among the classes,

there is another level of relationship expressed by using a special group of terms: properties. These property terms describe various features and attributes of the concepts, and they can also be used to associate different classes together. Therefore, the relationships among classes are not only superclass or subclass relationships, but also relationships expressed in terms of properties. Ontology has the following properties:

- It is domain specific.
- It defines a group of terms in the given domain and the relationships among them.

By clearly defining terms and the relationships among them, ontology encodes the knowledge of the domain in such a way that it can be understood by a computer. This is the basic purpose of ontology. RDF Schema and Web Ontology Language (OWL) are two popular languages for creating ontologies for semantic web.

2.2.2 Benefits of Ontology

Ontology provides a common and shared understanding about certain key concepts in the domain. They are:

- It provides a way to reuse domain knowledge.
- It makes the domain assumptions explicit.
- Together with ontology description languages (such as RDF schema), it provides a way to encode knowledge and semantics such that machines can understand.
- It makes automatic large-scale machine processing possible. Among all, fourth one is the factor that attracts the attention of the semantic web developers the most.

CHAPTER 3

ONTOLOGY IMPLEMENTATION

3.1 Implementation Pre-requisites

Semantic web can be considered as the replacement of the traditional web, which is currently being used. It is made by changing the data and documents into machine-readable data and documents on the web. For adding machine-readable description to the data and document in the semantic web, ontology is required. For ontology implementation, XML, RDF, OWL are the basic foundation.

3.2 XML

XML, stands for Extensible Markup Language, is a self-descriptive language for the transportation and storage of data, focusing on what data it is describing [**?**]. XML operates on two main levels: first, it provides syntax for document markup; and second, it provides syntax for creating vocabularies that can bring structure to both documents and data on the Web [**?**]. The XML syntax provides vendor independence, user extensibility, validation, human readability and the ability to represent complex structures. XML documents contain elements, attributes which are represented using tags. It follows several syntax rules:

- XML documents must have a root element which is the parent of all other elements.
- XML elements must have a closing tag.
- XML tags are case sensitive.
- XML elements must be properly nested.

It provides both programmers and document authors with a friendly environment. XML's rigid set of rules helps make documents more readable to both humans and machines. XML

tags are not predefined. As the name indicates, XML is extensible because it allows authors or developers to define their own tags and own document structures [**?**]. XML is hardware and software independent. So, it can be used on a wide variety of platforms and interpreted with a wide variety of tools.

3.3 RDF

RDF stands for Resource Development Framework and it is a standard recommended by W3C [**?**] for describing information or metadata of web resources so that computer application can easily understand. RDF is an application of XML. What XML is for syntax, RDF is for semantics [**?**], means it imposes necessary structural constraints to provide unambiguous methods of expressing semantics. RDF additionally provides a means for publishing both human-readable and machine-readable vocabularies designed to encourage the reuse and extension of metadata semantics among disparate information communities.

3.3.1 RDF Abstract Model

The key idea of RDF's abstract model is to break information into small pieces, and each small piece has clearly defined semantics so that machine can understand it and do useful things with it. Now, using RDF's terminology, a given small piece of knowledge is called a statement. Statement is formed combining the three elements resource, property and property value. This statement is divided into subject-predicate-object, which is known as triple. In RDF, information is represented by triple [**?**]. So, RDF statement has the following format :

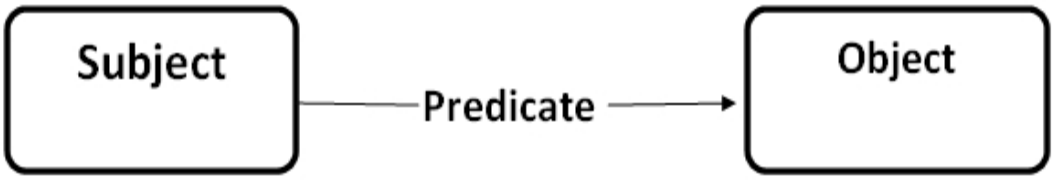


Figure 3.1: RDF Statement Model [8]

Where the subject and object are names for two things in the world, with the predicate being the name of a relation that connects these two things. Subject and object are also known to refer or denote things in real world. These things can be anything. They can be any

concrete thing or any abstract thing. These concrete or abstract things are called resources. Resources should be uniquely identified or in other word they should be unique. To do this we identify resources by Uniform Resource Identifier (URI). There are two kinds of URI, one that actually contains any resources and one that doesn't. The URI that contains actual resources is known as URL. Now it is not exactly necessary for the URI to actually contain any information or any resources. The main reason of using the URI is to differentiate various resources from each other. For example, to represent resources like an educational institute such as Military Institute of Science and Technology (MIST) in short form, one can use the full name while another can use the short form (MIST). Both names refer to the same institute. Though it is not possible for the machine to understand the similarity and the crawler will identify these two names as different institutes. But if these names were represented as a URI then for both name it would have been possible to employ a common URI. This is how using URI eliminates the conflicts on naming. Another important issue while using URIs is, whether we always invent new URIs or not. It is recommended that if an existing URI for identifying a resource is already found then it is better to use that URI instead of inventing a new URI.

3.3.2 URI

URI can be of two kinds. Hash URI and slash URI. Anyone of these can be used. Now to answer the question, which type of URI should be used or which type is more convenient, we refer to the fact that, using slash URI needs content negotiation mechanism, while using hash URI doesn't require any content negotiation mechanism.

3.3.3 Namespace

While using URI to describe a resource, often we see that all the resources have fairly long names. This is not quite convenient and not quite readable either. The solution to this issue is quite straightforward: a full URI is usually abbreviated by replacing it with its XML qualified name (QName). A QName contains a prefix that maps to a namespace URI, followed by a colon, and then a local name. For example,

http://nymaferdousmist.hostoi.com/department

http://nymaferdousmist.hostoi.com/faculties

These two URIs could be shortened by defining the prefix:-

Prefix	**Namespace**
MIST	http://nymaferdousmist.hostoi.com

And after that the URIs can be represented as:-

MIST:department

MIST:faculties

3.3.4 Predicates as URI

In a given RDF statement predicates denotes the relationship between the subject and object. In RDF abstract model it is desirable to represent the predicates using URIs instead of using string such as "is_a" or "has" etc.

Predicate in a RDF statement actually represents the property of a subject. Like instructor "resignation" in an institute.

3.3.5 RDF Syntax

A syntax representing the RDF model is required to store instances of this model into machine-readable files and to communicate these instances among applications. XML is this syntax. RDF imposes formal structure on XML to support the consistent representation of semantics [**?**]. Dublin core is also associated with this structure. Dublin core is a set of predefined properties for describing documents [**?**]. RDF uniquely identifies property-types by using the XML namespace mechanism. XML namespaces provide a method for identifying unambiguously the semantics and conventions governing the particular use of property-types by uniquely identifying the governing authority of the vocabulary.

3.3.6 RDF Schema

RDF schema declares valid vocabularies of RDF. It defines some valid application-specific classes, subclasses and properties [**?**]. RDF schema can add semantics to RDF predicates and resources: it defines the meaning of a given term by specifying its properties and what kinds ofobjects can be the values of these properties [**?**]. The core elements of RDF schema are as follows:
Core classes: rdfs:Resource, rdf:Property, rdfs:Class, rdfs:datatype
Core properties: rdfs:subClassOf, rdfs:subPropertyOf
Core constraints: rdfs:range, rdfs:domain

3.4 RDF Creation

As the world is proceeding with the development of semantic web, several software tools have been created to meet all the requirements needed for the semantic web. For the creation of the RDF, several tools are now available. RDF can be created using various tools like Reggie, PrismEd, DC-Dot, Eclipse.After the RDF document has been written, it can be checked by the RDF validator whether the document is valid or not. We preferred Eclipse for RDF creation.

3.5 RDF Crawling

A search engine's life starts from crawling. We shall include here the process of crawling. Clearly, before a search engine can tell us anything at all, it must know where everything is in advance. A crawler is used for this purpose. A URL server sends a list of URLs to the crawler for it to visit. This list of URLs is viewed as the seed URLs- the URLs that we want the crawler to start with. For each URL, the crawler downloads the Web document on this URL and finds all the hypertext links on that page that point to other Web pages. It then picks one of these new links and follows that link to download a new page, and finds more links on the new page until it decides to stop or there is no more links to follow. In semantic web, the search engine indexes these URLs in its indexation table. To make the indexation more comprehensive the search engine parses the string value of the RDF schema and uses them as keywords to index. So, it becomes easy to find out the document.

CHAPTER 4

IMPLEMENTATION OF SEMANTIC WEB TECHNOLOGY

4.1 RDF Generation

To implement semantic web technology at first we need to generate appropriate RDF documents. RDF may enable search engines and other tools for resource discovery to exchange and share metadata.

4.1.1 Metadata Elements

The classification process results in the production of a series of class marks appropriate to describe a particular document. However, the process can easily be used to pull out various other metadata elements. The most well known and well used metadata element set for resource discovery is Dublin Core. Compliance with a recognized standard is advisable because it encourages interoperability and consistency between applications. Dublin Core has evolved from the Digital Library community and consequently not all of its elements are as well suited to the automated search engine domain as those defined. There is, however a significant overlap and none of the Dublin Core elements are compulsory.

Elements used in Dublin Core(DC) [**?**]:

Element Name	Element Description
Creator	This element represents the person or organization responsible for creating the content of the resource; e.g., authors in the case of written documents
Publisher	This element represents the entity responsible for making the resource available in its present form; it can be a publishing house, a university department, etc.
Contributor	This element represents the person or organization not specified in a creator element who has made significant intellectual contributions to the resource but whose contribution is secondary to any person or organization specified in a creator element; e.g., editor, transcriber, illustrator
Title	This element represents the name given to the resource, usually by the creator
Subject	This element represents the topic of the resource; normally, it will be expressed as keywords or phrases that describe the subject or content of the resource
Date	This element represents the date associated with the creation or availability of the resource
Identifier	This element is a string or number uniquely identifies the resource; examples include URLs, Purls, ISBN, or other formal names
Description	This element is a free text description of the content of the resource; it can be a flexible format, including abstracts or other content descriptions
Language	This element represents the language used by the document
Format	This element identifies the data format of the document; this information can be used to identify the software that might be needed to display or operate the resource; e.g., postscript, HTML, text, jpeg, XML

Figure 4.1: Dublincore elements

4.1.2 RDF Generation using Dublin Core

DC Metadata tool will be found at http://www.ukoln.ac.uk/metadata/dcdot/. We shall submit the page www.mist.ac.bd. It will read the page and generate rdf.

Figure 4.2: Dublin Core Metadata Editor

4.1.3 RDF Generation using Development Tools

We can generate RDF documents using other development tools. There are many frameworks available for semantic web applications. These frameworks are often created for specific development domain and normally contain a set of common and reusable building blocks so that developers can use, extend, or customize for their specific business logic. With the help from such a framework, developers do not have to start from scratch each time an application is developed. More specifically, for development work on the Semantic Web, the main features of a framework may include the following:

- Core support for RDF, RDFS, and OWL;
- Inference capabilities for both RDFS ontologies and OWL ontologies;
- Support for SPARQL query;

- The handling of persistent RDF models, with the ability to scale efficiently to large datasets.
- It provides developers with the implementation of common tasks in the form of reusable code, therefore less repeated work and less bugs.
- It makes it easier to work with complex technologies such as the Semantic Web Technologies.
- It forces consistency within the team, even across platforms.
- It promotes design patterns, standards, and policies.

4.1.4 Current Tools on Semanticweb.org

The following tools are currently recorded. The most recently released tools are:

1. RDF2Go (Version 4.8.3 4 June 2013)

2. Bigdata (Version 1.2.3 31 May 2013)

3. Semantic Measures Library (Version 0.0.5 4 April 2013)

4. HermiT (Version 1.3.7 25 March 2013)

5. Fluent Editor (Version 2.2.2 20 March 2013)

Also there are many other tools currently available to be named.

Table 4.1: Available RDF Generation Tools

Name	Category	Version	Status	Released	License	By
AceRules	Category:Tool Category:Semantic Web tool		prototype			University of Zurich

AllegroGraph	Category:Tool Category:RDFstore Category:Reasoner	4.4.0.1	stable	12 January 2012	Pay Licensed Closed Source	Franz Inc
TopBraid EVN	Category:Tool Category:Semantic Web tool Category:Linked Data	1.2.0	stable	1 March 2013	Pay Licensed Closed Source	DIQA Project management GmbH
Jena .NET	Category:Tool Category:RDF store Category:Semantic Web development toolkit	0.3	beta	19 December 2010	BSD licenses	
HyperGraphDB	Category:Tool Category:RDF store	1.0	stable	22 April 2010	LGPL	
CubicWeb	Category:Tool Category:Semantic Web development toolkit	3.15.4	stable	18 September 2012	LGPL	Logilab
Bigdata	Category:Tool Category:RDF store	1.2.3	stable	31 May 2013	GPLv2	SYSTAP, LLC

OWLGrEd	Category:Tool Category:Ontology editor	1.0	stable	6 June 2011	Free	Institute of Mathematics and Computer Science, University of Latvia
Linked Media Framework	Category:Tool Category:Semantic Web tool Category:Semantic Web development toolkit Category:Semantic search tool Category:Linked Data Category:RDF store Category:RDF indexing service Category:Topic Linked Data	2.2.0	stable	6 July 2012	Apache License	Salzburg Research

4.2 Deployment of RDF Dcument on Web

Creating the RDF documents is not enough. To get it into work we need to upload these documents on the web. There are many issues regarding this task.

Dereferencing HTTP URIs: URI Dereferencing is the process of looking up a URI on the

Web in order to get information about the referenced resource. There are two approaches that data publishers can use to provide clients with URIs of information resources describing non-information resources: Hash URIs and 303 redirects.

Information Resources: When a URI identifying an information resource is dereferenced, the server of the URI owner usually generates a new representation, a new snapshot of the information resource's current state, and sends it back to the client using the HTTP response code 200 OK.

Non-Information Resources: Cannot be dereferenced directly. Therefore, Web architecture uses a trick to enable URIs identifying non-information resources to be dereferenced: Instead of sending a representation of the resource, the server sends the client the URI of a information resource which describes the non-information resource using the HTTP response code 303 See Other. This is called a 303 redirect. In a second step, the client dereferences this new URI and gets a representation describing the original non-information resource.

Content Negotiation:

HTML browsers usually display RDF representations as raw RDF code, or simply download them as RDF files without displaying them. This is not very helpful to the average user. Therefore, serving a proper HTML representation in addition to the RDF representation of a resource helps humans to figure out what a URI refers to. This can be achieved using an HTTP mechanism called content negotiation. HTTP clients send HTTP headers with each request to indicate what kinds of representation they prefer. Servers can inspect those headers and select an appropriate response. If the headers indicate that the client prefers HTML, then the server can generate an HTML representation. If the client prefers RDF, then the server can generate RDF.

Steps:

1. The client performs an HTTP GET request on a URI identifying a non-information resource. In our case a vocabulary URI. If the client is a Linked Data browser and would prefer an RDF/XML representation of the resource, it sends an Accept: application/rdf+xml header along with the request. HTML browsers would send an Accept: text/html header instead.

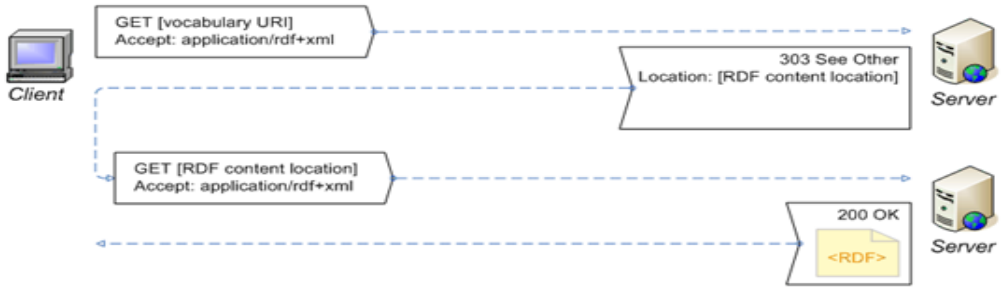


Figure 4.3: A complete example of a HTTP session for dereferencing a URI identifying a non-information resource

2. The server recognizes the URI to identify a non-information resource. As the server can not return a representation of this resource, it answers using the HTTP 303 See Other response code and sends the client the URI of an information resource describing the non-information resource. In the RDF case: RDF content location.

3. The client now asks the server to GET a representation of this information resource, requesting again application/rdf+xml.

4. The server sends the client a RDF/XML document containing a description of the original resource vocabulary URI.

4.3 Indexing URL in Semantic Web Search Engine

After deployment, URL of the website that links these documents needed to be indexed in a semantic web search engine. By doing these after a certain period of days the search engine is able to find the documents. Swoogle adapts the Sire, a custom indexing and retrieval engine. It employs a TF/IDF model with a standard cosine similarity metric. It indexes discovered documents by using either character N-Gram or URIrefs as keywords to find relevant documents and to compute the similarity among a set of documents. Traditional IR

techniques have the advantage of being faster, while taking asomewhat more coarse view of the text. They can thusquickly retrieve a set of SWDs that deal with a topic basedon text similarity alone.In addition to the efficiency, there are a number of reasonswhy one would want to apply IR techniques to this problem. For one thing, documents are not entirely markup. We would like to be able to apply search to both the structured and unstructured components of a document. Relatedto this point, it is conceivable that there will be some text documents that contain embedded markup. In addition, wemay want to make our documents available to commonlyused search engines, such as Google. This implies that thedocuments must be transformed into a form that a standard IR engine can understand and manipulate. IR techniques also have some value characteristics, including wellresearched methods for ranking matches, computing similarity between documents, and employing relevance feedback. Traditional IR techniques look at a document as eithera collection of words or N-Gram. An N-Gram is an n-character segment of the text which spans inter-word boundaries. The N-Gram approach is typically employed by sliding a window of n-characters along the text, and taking asample at each one character step. The use of N-Gramscan result in a larger vocabulary, as single words can contain multiple N-Grams.

4.3.1 Advantages of N-Grams

1. Inter-word relationships are preserved, where they aretypically not in word based approaches.

2. Resistant to errors.

3. Treatment of URIs as terms.

Given a set of keywords defining a search, we may want to match documents that have URIs containing those keywords. For example, consider a search for ontologies for “time”. The search keywords might be time temporal interval point before after during day month year eventually calendar clock durations end begin zone. Search results might include documents containing URIs such as:

http://foo.com/timeont.owltimeInterval

http://foo.com/timeont.owlCalendarClockInterval

http://purl.org/upper/temporal/t13.owltimeThing

Clearly, exact matching based on words only would miss these documents (based on the URIs given). However, N-Grams would find a number of matches.

4.4 Retrieval of RDF Document

To test whether the RDF creation and search engine indexing is successful; we try to retrieve the RDF documents by searching. If the indexing is successful it will be able to find the RDF.

Optimized Linear Instance Retrieval:

One possible alternative is to consider one individual at a time. Hence, the procedure instance retrieval(Cq,A) can be implemented by using the following procedure call: linear instance retrieval(Cq, contract(i,A), individuals(A)) where individuals(A) returns the set of individuals mentioned in the A-box A and the function contract computes a transformation of an A-box w.r.t. an individual. The idea is to transform tree-like role assertions "starting"from the individual i into equisatisfiable concept assertions with existential restrictions.

Algorithm 1 *linear_instance_retrieval*$(C, A, candidates)$:

```
result := {}
for all ind ∈ candidates do
  if instance?(ind, C, A) then
    result := result ∪ {ind}
return result
```

Figure 4.4: Algorithm for Optimized Linear Instance Retrieval

Binary Instance Retrieval:

Instance retrieval(Cq,A) is implemented by calling the procedure binary instance retrieval(Cq, contract(i,A), individuals(A)). The function partition divides a set into two partitions. Given the partitions,binary instance retrieval calls the function partition instance retrieval. The idea of partition instance retrieval is to first check whether none of the individuals in a partition is an instance of the query concept C.

Algorithm 4 *binary_instance_retrieval*$(C, A, candidates)$:

```
if candidates = ∅ then
  return ∅
else
  (partition1, partition2) := partition(candidates)
  return partition_instance_retrieval(C, A, partition1, partition2)
```

Algorithm 5 *partition*(s): /* $s[i]$ refers to the i^{th} element of the set s */

```
if |s| ≤ 1 then
  return (s, ∅)
else
  return ({s[1], ..., s[⌊n/2⌋]}, {s[⌊n/2⌋ + 1], ..., s[n]})
```

Figure 4.5: Algorithm for Binary Instance Retrieval

Dependency Based Instance Retrieval: Although binary instance retrieval is found to be faster in the average case, one can do better. Dependency based instance retrieval is always faster than binary search retrieval.

Static Index-based Instance Retrieval:

The standard way to compute the index is to compute the direct types for each individual mentioned in the A-box separately (one-individual-at-a-time approach). In order to compute the direct types of individuals w.r.t. a T-box and an A-box, the T-box must be classified, i.e., for each concept name mentioned in the T-box (and the A-box)the most-specific subsumers (function parents) and least-specific subsumees (functionchildren) are precomputed. Thus, parents and children are not really queries but just functions accessing results stored in data structures.Another view is that the children(or parents) relation defines a lattice whose nodes are concept names.

Dynamic Index-based Instance Retrieval:

In this algorithm, In this new approach, the function associated inds(C) returns an individual i even if C is not “most specific”, i.e. even if there might exist a subconcept D of C such that i is also an instance of D. If an individual I is found not to be an instance of a query concept D, this is recorded appropriately by including i in associated non instance(D) if there is no

Algorithm 8 *static_index_based_instance_retrieval*(C, A):

if $\exists N \in CN : N \in synonyms(C)$ **then**
  return $\bigcup_{D \in descendants(C)} associated\_inds(D)$
else
  $known\_results := \bigcup_{D \in descendants(C)} associated\_inds(D)$
  $candidates := \bigcup_{P \in parents(C)} (\bigcup_{D \in descendants(P)} associated\_inds(D))$ (*)
  return $known\_results \cup instance\_retrieval(C, A, candidates \setminus known\_results)$

Figure 4.6: Algorithm for Static Index-based Instance Retrieval

E 2 ancestors(D) such that i 2 associated non instance(E) (non-redundant caching). The non-instances of aquery concept can then be discarded from the set of candidates.

Algorithm 11 *dynamic_index_based_instance_retrieval_1*(C, A):

$known\_results := \bigcup_{D \in descendants(C)} associated\_inds(D)$
$possible\_candidates := \bigcup_{D \in (ancestors(C) \setminus \{C\})} associated\_inds(D)$
$candidates := possible\_candidates \setminus \bigcup_{D \in ancestors(C)} associated\_non\_instances(D)$
return $known\_results \cup instance\_retrieval(C, A, candidates \setminus known\_results)$

Figure 4.7: Algorithm for Dynamic Index-based Instance Retrieval

CHAPTER 5

SEMANTIC WEB IMPLEMENTATION USING SOFTWARE TOOLS

5.1 Acquisition of Necessary Tools and Packages

RDF document is the basic building block of semantic web service. To create RDF document efficiently we use software tools. For example we use eclipse IDE and Jena framework which is a semantic web framework for java. To bring eclipse and Jena into work at first we have to link Jena package with eclipse. This way we can write java code using Jena library to effectively generate required RDF document as output.

5.1.1 Jena

Jena is a free, open-source Java platform for applications on semantic web.Jena is now believed to be the most used Java toolkit for building applications on the Semantic Web. Jena's comprehensive support for Semantic Web application development is quite obvious, given its following components:

- An RDF API;
- An OWL API, which can also be used as RDFS API;
- Reading and writing RDF in RDF/XML, N3, and N-triples formats;
- In-memory and persistent storage of RDF models;
- SPARQL query engine, 4cmRule-based inference engine.

Getting Jena Package to access Jena, we need to go to the following page:
http://jena.sourceforge.net/ This links to the home site of Jena, as shown in Figure:

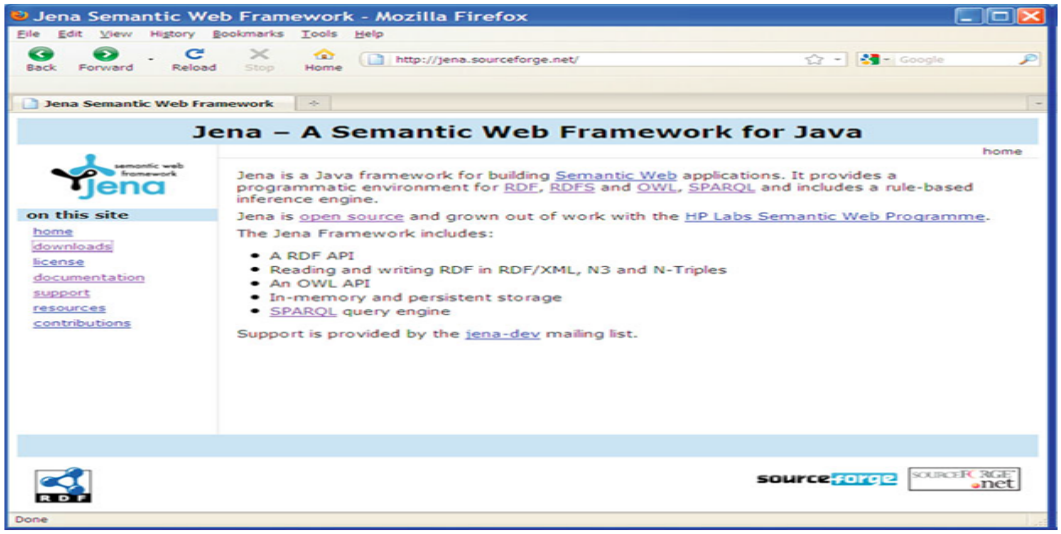


Figure 5.1: Jena Home Page

To download Jena package we need to click on download link on the left pane of this home-page. After that we should chose the latest version of Jena package. Once the download is done we will find a zip file named Jena-2.10.* in our local hard drive. Then unzipping that file will create a Jena package directory in our hard drive [**?**].

5.2 Linking Jena Package with Eclipse

At first we need to set the 'classpath' variable from the 'environmental variable' of our machine with the path location of Jena home directory. In order to create RDF document, eclipse IDE should include Jena package as a library. This way all the library functions written in java code to create xml vocabularies and properties for RDF document can be recognized by the IDE.

As far as Eclipse is concerned, the difference between a plain Java project and a Java project that uses Jena library is that Eclipse has to know where to find the Jena library files that the project refers to. Once it can locate the library files, it will be able to load the related class definitions from the library as the necessary supporting code to our project.

One way to accomplish this is to create a lib directory in our project workspace, copy Jena related library files to this lib directory, and then add this lib directory into our project's build path. This will work; however, a user library is a better solution to use. In Eclipse, a user library is a user-defined library (a collection of jar files) that one can reference from any project. In other words, once we have configured a user library, we can use it in multiple

different projects. Furthermore, if Jena releases a new updated version, updating the user library once will guarantee that all the projects using this library will all see the newly updated version. If we had created a library under each specific project workspace, we would have to copy the new version to every workspace that makes use of Jena. To configure a user library, we need to open up Eclipse and select Window from the menu bar. From the drop-down menu list, select preferences, which will bring up the Preferences dialogue window. In this window, then open Java on the left navigation tree and open Build Path. Once Build Path is open, we need to select User Libraries as shown in Figure:-

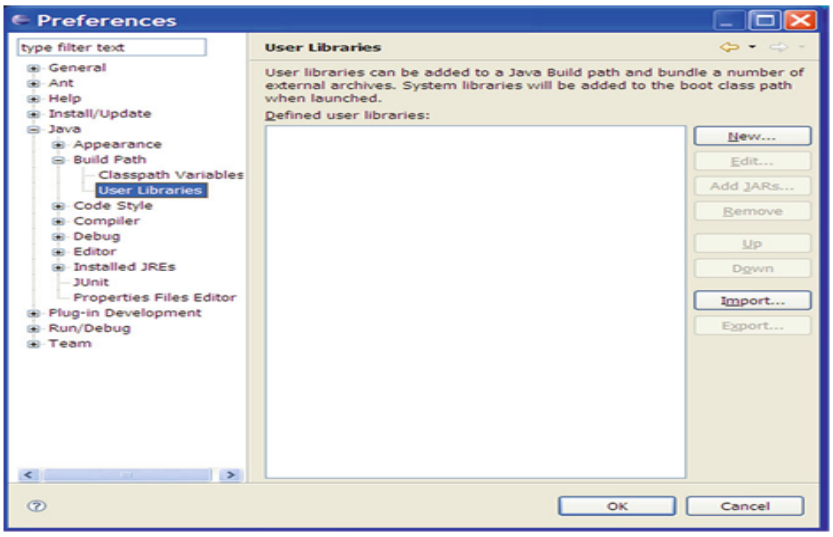


Figure 5.2: Preference Dialogue Window [8]

5.3 Java Codes to Generate RDF

To generate the required RDF documents, we write java code to automatically create the appropriate RDF for us. Different Jena library functions perform this task. We write several java codes to create various RDF documents for an educational institution.

5.4 RDF Verification

To test if the generated RDF documents are indeed correct and working we test these documents in RDF validator.

We went to the following page for RDF verification:

http://www.w3.org/RDF/Validator/

5.5 Deployment of RDF Documents on Web Server

For deployment, we need to buy a domain name. For our purpose, here we first went to a free domain service at http://www.000webhost.com. Then we registered a domain named nymaferdousmist.hostoi.com. For file transfer, we used FTP FileZilla Client. Then we created a RDF document named misthome.rdf for uploading.

In FileZilla, to connect to a site we have to fill the username and password of that site.

Username: a4629868

Port: 21

Then a successful connection will set up and all directory will be listed. We uploaded misthome.rdf file to the site nymaferdousmist.hostoi.com.

To upload a file with FileZilla, Right click on the file and Click upload. Then we need to use SameAs tool to find the file. In sameAs, we write

http://nymaferdousmist.hostoi.com/misthome.rdf

Then equivalent link will be found. If we click on the file, then we get our rdf file misthome.rdf. The process described below with figures:

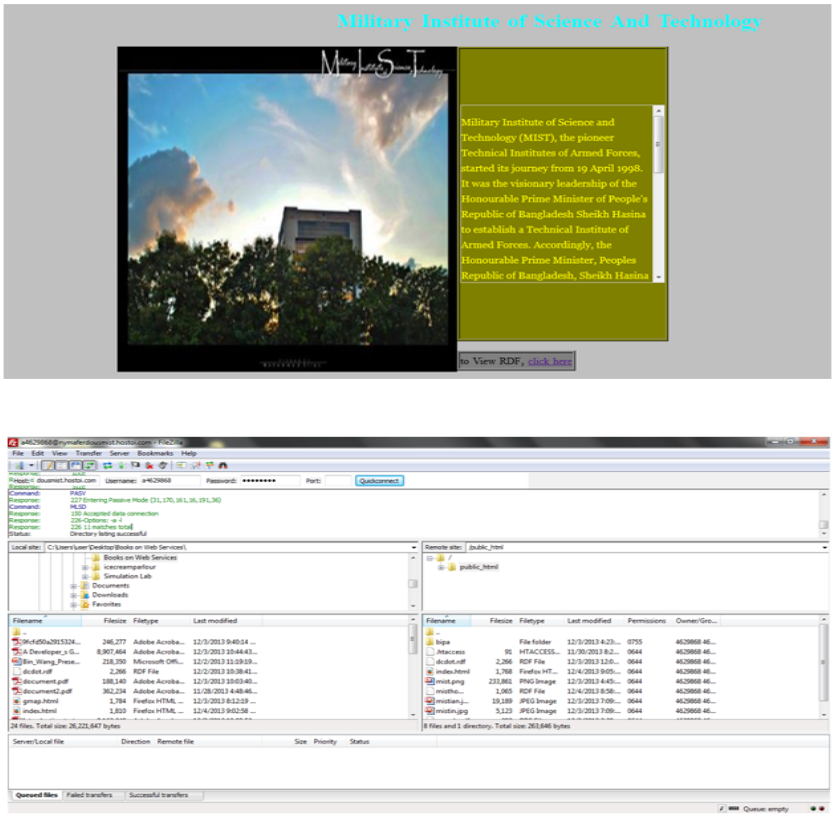
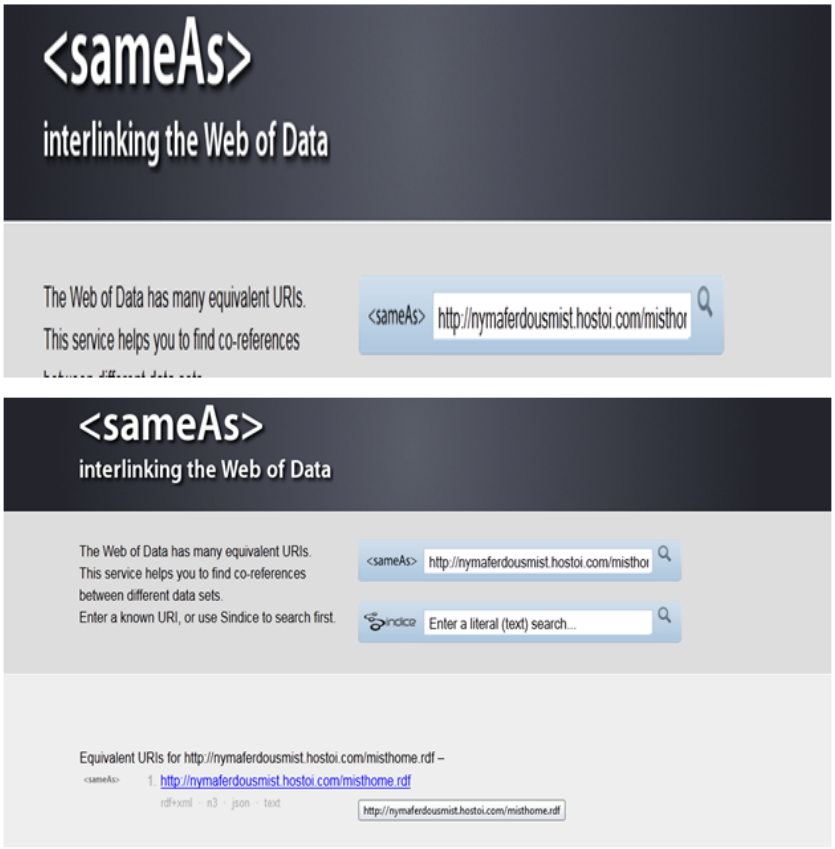


Figure 5.3: File Upload using Filezilla



This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<rdf:RDF>
 -<rdf:Description rdf:about="http://www.mist.ac.bd/rdf">
    <home:title> Military Institute of Science And Technology </home:title>
    <home:author> Nyma,Sweety,Noman </home:author>
    <home:homepage> http://mist.ac.bd </home:homepage>
   -<home:objective>
      To establish a prestigious academic institute for studies in different fields of engineering and technology for military personnel and civil officials/ students of home and abroad at degree and post
      graduate levels.
    </home:objective>
   -<home:courses>
     -<rdf:Bag>
        <rdf:li>CSE</rdf:li>
        <rdf:li>EEE</rdf:li>
        <rdf:li>CE</rdf:li>
        <rdf:li>ME</rdf:li>
        <rdf:li>AERO</rdf:li>
        <rdf:li>NAME</rdf:li>
      </rdf:Bag>
    </home:courses>
    <home:email>info@mist.ac.bd </home:email>
    <home:address> Mirpur Cantonment, Dhaka-1216, Bangladesh.</home:address>
  </rdf:Description>
</rdf:RDF>
```

Figure 5.4: Document Deployment Process

5.6 URL Indexing in swoogle Search Engine

To get these RDF documents about an educational institution we use swoogle as a semantic web search engine. But for Swoogle to find these documents when crawling, URL of the website should be first submitted in Swoogle. For that purpose we first go to the following page:

http://swoogle.umbc.edu/

Then click submit-url on the bottom of this page tosubmit url of of our created website by navigating to the page as shown in figure below:

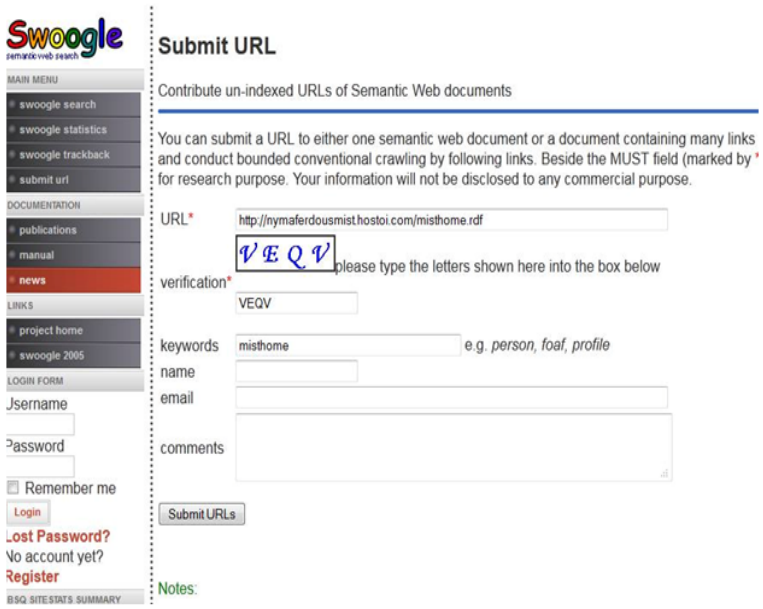


Figure 5.5: URL Submission

CHAPTER 6

CONCLUSION AND DISCUSSION

The thesis is finally concluded in this chapter. This chapter emphasizes on the challenges that we have met during implementation and the future works that can be done based on the thesis.

6.1 Implementation Challenges

To access a web server for uploading the website and RDF document we need to pay for domain and hosting. Again a search engine doesn't find the appropriate documents easily. It takes some time for the search engine to index these documents. Also we used a tool <sameAs> that interlinks the web of data. If there is any error in the rdf file, then it cannot be deployed. These challenges can be described as follows:

6.1.1 RDF Deployment Issue

For the generated RDF to be deployed on the web we need to first take care of some issues. Like, our generated RDF documents should have to be uploaded on any website. This website then has to be deployed on the web. For that purpose first we need a domain name. Then to access that domain we need hosting. That means our website should be placed in any server. But for these purpose we need to buy a domain name and hosting for our website. To get over that limitation we used free hosting service from www.000webhost.com. And then we used free domain for our website too.

6.1.2 Document Indexation on Search Engine

Generating and deploying RDF documents on the web is not enough. We have to make sure that the created RDF is indeed correct and can be found when searched. For this we submit our website link to Swoogle for indexing. But Swoogle can't index a document just after the URL of that document has been submitted. For Swoogle to index the document and create an entry on its index table it takes some times. So, we can't be sure immediately whether our effort to find the RDF document when searched is indeed successful.

6.2 Future Development

As for now we created the RDF documents first, and then uploaded these documents manually into the server. In future we will try to automate this process. Again the number of RDF documents we created for an educational institution is not enough for getting the full benefit of semantic web. We will try to create many more RDF documents to make it more convenient.

6.3 Benefits

1. Current web search engines such as Google and all the web do not work well with documents encoded in the semantic web languages RDF and OWL. These retrieval systems are designed to work with natural languages and expect documents to contain unstructured text composed of words. They do a poor job of tokenizing semantic web documents; they do not understand conventions such as those involving XML namespace. Moreover, they do not understand the structural information encoded in the documents and are thus unable to take advantage of it. But, our created RDF files can help to search mist homepage semantically.

2. By creating as much RDF documents for an educational institution we make it easier for the semantic web to find the required data we are looking for. Understanding the semantics of those data, integrating and correlating different activities to automate the process will be more effective and efficient.

APPENDIX A

XML EXAMPLE

Here is a simple example of an XML document which is a breakfast food menu of a restaurant:

<?xml version="1.0" encoding="ISO-8859-1" ?>
<breakfast_menu >
<food >
<name>Belgian Waffles</name >
<price>$5.95 </price>
<description >Two of our famous Belgian Waffles with plenty of real maple syrup </description >
<calories>650 </calories>
</food>
<food >
<name >Strawberry Belgian Waffles </name>
<price >$7.95 </price >
<description>Light Belgian waffles covered with strawberries and whipped cream </description>
<calories>900</calories >
</food>
</breakfast_menu>

APPENDIX B

GENERATED RDF USING DUBLIN CORE

Generated RDF:

<?xml version= “1.0”?>

<!DOCTYPE rdf:RDF SYSTEM “http://dublincore.org/documents/2002/07/31/dcmes-xml/dcmes-xml-dtd.dtd”>

<rdf:RDF

xmlns:rdf=“http://www.w3.org/1999/02/22-rdf-syntax-ns”

xmlns:dc=“http://purl.org/dc/elements/1.1/”

<rdf:Description rdf:about=“http://mist.ac.bd/”>

<dc:title>

Military Institute of Science and Technology (MIST)

</dc:title>

<dc:creator>

</dc:creator>

<dc:subject>

NAVAL; EXAM; Student; TECHNOLOGY; NOTICE; Cirtificate;

COMMUNICATION; M.SC; News; Board; Routine; Sc/M; Solutions;

M.Sc./M; ICEEICT; Engineering; STUDENT; GSO-1; Coord;

Based; Computer; ELECTRICAL; BUET; Undergraduate; Class;

Conferences; Electronic; CATS; Director; Asst; Mechanical;

B.Sc; Graduates; Notice; Design; Location; Welcome;

Seminars; Conf; Commandant; ENGINEERING; REGISTER; Photo;

Friday; Table; Science; AE; Foreword; Colonel; Electrical;

Home; Library; Login; OIC; HQ; 1:30pm; Programme; Read;

admission; Hum; PROG-OCT; System; Job; Librarian; Micro-Controller; Admission; Journal; List; Prospectus-2013; NOGOR; Research; Circular; Expansion; requested; Charges; conduct; Migration; Programe; Exam; M.Sc/M; Site; Examinees; Opportunity; Eligible; R&D; Postgraduate; WEBMAIL; View; Alumni; Department; Technology; ROUTINE; Objectives; Aim; Related; PROGRAM; Album; Communication; Halls; Embedded; Future; CE; Program; Details; Links; SEMSTER; TENDER; Newsletter; Fees; PIMS; Online; DU; Registration; Affiliation; Departments; MIST; Architechture; Marine; DAAQMG; Advancement; Instruction; Map; Contents; NAME; Database; Gallery; CIVIL; GSO-2; INASP; Code; Students; Candidates; Civil; Development; Policy; ENGG; BUP; BanglaJOL; GSO; Contact; test; rescheduled; International; November; CSE; Editorial; Dean; Downloads; Course; Staff; Sc; DATABASE; WKSP; Aeronautical; ACCREDITATION; Projects; Capabilities; Admin; CEESR; Engg; Admission-2013; INFORMATION; Various; EECE; Dept; Humanities; Academic
</dc:subject>
<dc:description>
mist home page
</dc:description>
<dc:date>
12-2-2013
</dc:date>
<dc:type>
Text
</dc:type>
<dc:format>
text/html
</dc:format>

<dc:format>
31617 bytes
</dc:format>
</rdf:Description>
</rdf:RDF>

APPENDIX C

GENERATED RDF DOCUMENTS FOR AN EDUCATIONAL INSTITUTION

Generated RDF:

RDF for the Departments:

<?xml version= "1.0"?>

<!DOCTYPE rdf:RDF SYSTEM "http://dublincore.org/documents/2002/07/31/dcmes-xml/dcmes-xml-dtd.dtd">

<rdf:RDF

xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns"

xmlns:dc="http://purl.org/dc/elements/1.1/"

<rdf:Description rdf:about=":://mist.ac.bd/departments.php?content_id=7935112">

<dc:title>

Military Institute of Science and Technology (MIST)

</dc:title>

<dc:subject>

NAVAL; EXAM; Student; NOTICE; View; Board; Alumni;

Department; Solutions; Technology; Engineering; GSO-1;

Objectives; Aim; Coord; Album; Halls; Communication;

Future; CE; Computer; Details; Program; Newsletter; TENDER;

Fees; Undergraduate; PIMS; Conferences; Electronic; CATS;

Director; Asst; Affiliation; Departments; Architechture;

MIST; Marine; DAAQMG; Advancement; Mechanical; Map;

Location; Seminars; Contents; NAME; Commandant; Database;

Gallery; INASP; GSO-2; Photo; Code; Science; Table; Civil;

Development; AE; Policy; Foreword; BanglaJOL; Colonel; GSO;
Electrical; Contact; Home; Login; Library; OIC; HQ; CSE;
Editorial; Hum; Dean; Downloads; Staff; Job; Librarian;
Admission; Sc; Journal; NOGOR; WKSP; Research;
Aeronautical; Projects; Expansion; Capabilities; Admin;
Charges; conduct; Admission-2013; Various; EECE; Exam;
Site; Opportunity; Humanities; Academic; RD; Postgraduate

</dc:subject>
:publisher>
</dc:publisher>

<dc:type>
Text
</dc:type>
<dc:format>
text/html
</dc:format>
<dc:format>
18084 bytes
</dc:format>
</rdf:Description>
</rdf:RDF>

RDF for the Feculties:

<?xml version= “1.0”?>
<!DOCTYPE rdf:RDF SYSTEM “http://dublincore.org/documents/2002/07/31/dcmes-xml/dcmes-xml-dtd.dtd”>

<rdf:RDF
xmlns:rdf=“http://www.w3.org/1999/02/22-rdf-syntax-ns”
xmlns:dc=“http://purl.org/dc/elements/1.1/”
<rdf:Description rdf:about=“http://mist.ac.bd/common_content_dept.php?content_id=7591522&content_id_dept=c31952070”>

<dc:title>
Military Institute of Science and Technology (MIST)
</dc:title>
<dc:subject>
NAVAL; Amir; EXAM; Student; NOTICE; News; Board; Solutions;
Engineering; GSO-1; Muzakkir; Cdr; Coord; Shazzadul; Md;
Major; Computer; Download; Saif-ul; Undergraduate;
Electronic; Director; Asst; Mechanical; Location;
Commandant; Captain; Maowa; Lt; Table; Science; AE;
Foreword; Colonel; Electrical; Home; Library; OIC; HQ;
Marzia; Hum; Syed; Librarian; Dr; Admission; Sigs; Lazima;
Khan; Journal; Mahmud; Hossain; NOGOR; Research; Shamsul;
Ahmed; Faculty; Ex; Expansion; Maintenance; Charges;
conduct; Exam; Alam; Site; news; Karim; R&D; Postgraduate;
Anisur; View; Professor; Department; Technology;
Objectives; Afzal; Aim; Jannatul; Communication; Assistant;
Future; CE; Kazi; Program; Details; TENDER; Newsletter;
Fees; PIMS; Affiliation; MIST; Architechture; Marine;
DAAQMG; Nazia; Advancement; Engineer; psc; Map; Contents;
NAME; Database; Lieutenant; GSO-2; Head; INASP; Code;
Civil; Wali; Development; Policy; BanglaJOL; GSO; Contact;
Mohammad; Taher; Fahim; Azmal; CSE; Facilities; Editorial;
Dean; Ansari; Staff; Sultana; Hasan; Mollah; Azam; Majadi;
Sc; Abdullah; Rahman; Islam; Lecturer; WKSP; Aeronautical;
Abu; Achievements; Sharifa; Capabilities; Admin; Admission- 2013; te;
Various; EECE; Mahboob; Dept; Humanities; Nazifa;
Academic; Rania

</dc:subject>
:publisher>
</dc:publisher>

<dc:type>

Text

</dc:type>

<dc:format>

text/html

</dc:format>

<dc:format>

43692 bytes

</dc:format>

</rdf:Description>

</rdf:RDF>

RDF for other facilities:

<?xml version= “1.0”?>

<!DOCTYPE rdf:RDF SYSTEM “http://dublincore.org/documents/2002/07/31/dcmes-xml/dcmes-xml-dtd.dtd”>

<rdf:RDF

xmlns:rdf=“http://www.w3.org/1999/02/22-rdf-syntax-ns”

xmlns:dc=“http://purl.org/dc/elements/1.1/”

<rdf:Description rdf:about=“http://mist.ac.bd/common_content_dept.php?content_id=7591522&content_id_dept=c435631321”>

<dc:title>

Military Institute of Science and Technology (MIST)

</dc:title>

<dc:subject>

NAVAL; EXAM; Student; NOTICE; View; News; Board; Training;

Department; Solutions; Technology; Engineering; GSO-1; Lab;

Objectives; Aim; Coord; Communication; Network; Future; CE;

Computer; PC; Download; Program; Details; Newsletter;

TENDER; Fees; Undergraduate; PIMS; ControllerLab;

Electronic; Director; Central; Affiliation; Asst;

Architechture; MIST; Marine; DAAQMG; Advancement;

Mechanical; Map; Location; Digital; Contents; NAME;

Commandant; Database; INASP; Head; GSO-2; Internet; Code;
Table; Science; Civil; Development; AE; Policy; Foreword;
BanglaJOL; Colonel; Contact; GSO; Server; Electrical;
Browsing; Home; Library; OIC; HQ; Facilities; CSE;
Editorial; Dean; Hum; Staff; Librarian; Micro; Admission;
Sc; Journal; NOGOR; WKSP; Research; Aeronautical;
Achievements; Ex; Faculty; Expansion; Intelligence/VLSI;
Capabilities; Admin; Charges; conduct; Artificial;
Admission-2013; Industrial; Various; Microprocessor; EECE;
Exam; Site; Software; Humanities; Cell; news; Academic;
RD; Postgraduate

</dc:subject>
<dc:publisher>
</dc:publisher>

<dc:type>
Text
</dc:type>
<dc:format>
text/html
</dc:format>
<dc:format>
46691 bytes
</dc:format>
</rdf:Description>
</rdf:RDF>