SEMANTIC WEB : DATA REPRESENTATION IN A DISTRIBUTED ENVIRONMENT

AZHAR BIN TAJUL NOOR

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2008

SEMANTIC WEB : DATA REPRESENTATION IN A DISTRIBUTED ENVIRONMENT

AZHAR BIN TAJUL NOOR

DISSERTATION SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF INFORMATION TECHNOLOGY

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2008

ABSTRACT

This thesis discusses the development of specific tools such as XML–SOAP-based system towards the realization of the Semantic Web. Semantic Web can be seen as a huge engineering solution, once it becomes easier to publish data in a repurposable form. In this thesis, the Semantic Web discussion is generally involved on syntaxes which use to represent data. As implementing the Semantic Web requires adding semantic metadata to the information resources, XML-SOAP-based system has paved the road by adding some metadata in the form of human-readable tags that describe data. In addition, XML documents can include information about the author of a web page, relevant keywords for search engine optimization, and the software tools used to create the XML file. This will allow machines to effectively process the data based on the semantic information that describes it.

ACKNOWLEDGEMENT

First and foremost, I would like to express the greatest gratitude to my thesis supervisor, Prof. Dato' Ir. Dr. Mashkuri Haji Yaakob for his brilliant guidance, encouragement, constructive feedback, resourceful ideas and suggestions that helped in enhancing my thesis.

My heartfelt gratitude also goes to all staff in the Faculty of Computer Science and Information Technology (FCSIT) University of Malaya for their supports and words of encouragement.

I would also like to thank my beloved family members for their love and support. To my wife, Rozita Md. Ezadin and my son, Luqman Hadi, their understanding always becomes my motivator along the path of accomplishing this thesis.

Most of all, I am grateful to Allah S.W.T, for all the blessings in my life.

TABLE OF CONTENTS

ABBREVIATIONS

API	Application Programming Interface
B2B	Business-to-Business
CERN	*Conseil Europeen pour le Recherche Nucleaire* (European Laboratory for Particle Physics)
DAML	DARPA Agent Markup Language
DCOM	Distributed Component Object Model
DTD	Document Type Definition
FTP	File Transfer Protocol
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
OIL	Ontology Inference Layer
RDF	Resource Description Framework
SOAP	Simple Object Access Protocol
SW	Semantic Web
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
WAIS	Wide Area Information Server
WSDL	Web Services Description Language
WWW	World Wide Web
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language
XSLT	eXtensible Stylesheet Language Transformation

LIST OF FIGURES

LIST OF TABLE

1.0 INTRODUCTION

1.1 Overview

The web, once solely a repository for text and images, is evolving into a provider of services - *information-providing services,* such as flight information providers, temperature sensors, and cameras, and *world-altering services*, such as flight booking programs, sensor controllers, and a variety of e-commerce and business-to-business (B2B) applications. In the next decade, computers will most likely be ubiquitous, and most devices will have some sort of computer inside them.

Today's web was designed primarily for human interpretation and use. Nevertheless, we are seeing increased automation of web service interoperation, primarily in B2B and e-commerce applications. Generally, such interoperation is realized through APIs that incorporate hand-coded information-extraction code to locate and extract content from the HTML (HyperText Markup Language) syntax of a web page presentation layout. Unfortunately, when a web page changes its presentation layout, the API must be modified to prevent failure. Fundamental to having computer programs or agents implement reliable, large-scale interoperation of web services is the need to make such services computer interpretable - to create a semantic web of services whose properties, capabilities, interfaces, and effects are encoded in an unambiguous, machine-understandable form.

The realization of the Semantic Web (SW) is underway with the development of new AI (Artificial Intelligence) inspired content markup languages, such as OIL, DAML+OIL, DAML-L and XML. These languages have a well-defined semantics and enable the markup and manipulation of complex taxonomic and logical relations between entities on the web. A fundamental component of the semantic web will be the markup of web services to make them computer-interpretable, use-apparent and agent-ready.

Through the billions of web pages created with HTML, or generated dynamically by underlying web database service engines, the web captures almost all aspects of human endeavor and provides a fertile ground for data mining. However, searching, comprehending and using the semi structured information stored on the web poses a significant challenge because this data is more sophisticated and dynamic than the information that commercial database systems store.

Furthermore, semantic web is needed with a well-established mechanism to express information that is machine-interpretable and allows syntactic and semantic interoperability among web applications. Although XML (eXtensible Markup Language) and RDF (Resource Description Framework) offer foundations for, respectively, syntactic and semantic interoperability, their mechanisms cannot accomplish this goal. XML by itself will let the same semantic unit be expressed in more than one syntactic structure. XML, RDF, and RDF Schema combinations might solve this multiple-structures problem, but they would still lack expressive power. For example, conditions and constraints could not be specified.

1.2 Problem Statement

The web presents the user with documents, called web pages, full of links to other documents or information systems. Selecting one of these links, the user can access more information about a particular topic. Web pages include text as well as multimedia (images, video, animation, sound). Servers are connected to the internet to allow users to traverse ("surf") the web using a web browser. More than that, the World Wide Web (WWW) is a vast repository of information. The great success of the current WWW leads to a new challenge, a huge amount of data is interpretable by humans only, and machine support is limited. There are now several billion documents on the WWW, which are used by more than 300 million users globally and millions more pages on corporate intranets. The continued rapid growth in information volume makes it increasingly difficult to find, organize, access and maintain the information required by users.

On the other hand, the current web also widely exploited many-to-many data-interchange medium and it poses new requirements for any exchange format:

- **Universal expressive power.** Because it is not possible to anticipate all potential uses, a web-based exchange format must be able to express any form of data.
- **Syntactic interoperability.** Applications must be able to read the data and get a representation that can be exploited. Software components like parsers or query APIs, for instance, should be as reusable as possible among different applications. Syntactic interoperability is high when the parsers and APIs needed to manipulate data are readily available.

- **Semantic interoperability.** One of the most important requirements for an exchange format is that data be understandable. Whereas syntactic interoperability is about parsing data, semantic interoperability is about defining mappings between terms within the data, which requires content analysis.

Thus, the notion of a semantic web that provides enhanced information access based on the exploitation of machine-processable metadata has been proposed.

1.3 Motivation of the Study

The semantic web began at the end of the last century as an object of investigation and experimentation. It has grown into a subject of huge interest to several research communities and to web users at large. Many people consider it the next generation of the web, a web that machines themselves can handle to help users.

Furthermore, data mining will play an important role in semantic web "intelligence" because the web is current incarnation still cannot provide high-quality, intelligent services. Several factors contribute to this problem and motivate the research.

1. **Lack of high-quality keyword-based searches**. The quality of keyword-based searches suffers from several inadequacies:

   - A search often returns many answers, especially if the keywords posed include terms drawn from perennially popular categories such as sports, politics, or entertainment;
   - Overloading keyword semantics can return many low-quality answers - for example, depending on the context, a *jaguar* could be an animal, car, sports team, or computer; and
   - A search can miss many highly related pages that do not explicitly contain the posed keywords, for example, a search for the term *data mining* can miss many highly regarded machine learning or statistical data analysis pages.
   - Incorporating data semantics could substantially enhance the quality of keyword-based searches.

2. **Lack of automatically constructed directories.** A topic or type-oriented web information directory presents an organized picture of a web sector and supports semantics-based information searches, which makes such a directory highly desirable. For example, following hierarchical links like *Malaysia* > *universities* > *computer science* > *graduate program* makes searches more efficient. Unfortunately, developers must construct such directories manually. Even then, these costly directories provide only limited coverage and developers cannot easily scale or adapt them.

3. **Lack of effective deep-web access.** In July 2000, analysts estimated that searchable databases on the web numbered at least 100,000. These databases provide high-quality, well-maintained information, but are not effectively accessible. Because current web crawlers cannot query these databases, the data they contain remains invisible to traditional search engines. Conceptually, the deep web provides an extremely large collection of autonomous and heterogeneous databases, each supporting specific query interfaces with different schema and query constraints. To effectively access the deep web, we must integrate these databases.

4. **Lack of semantics-based query primitives.** Most keyword-based search engines provide a small set of options for possible keyword combinations, essentially "with all the words" and "with any of the words." Some web search services, such as Google and Yahoo, provide more advanced search primitives, including "with exact phrases", "without certain words" and with restrictions on date and domain site type.

5. **Lack of feedback on human activities.** Humanity collective behavior often provides the best teacher. Web page authors provide links to "authoritative" web pages and also traverse those web pages they find most interesting or of highest quality. Unfortunately, while human activities and interests change over time, web links may not be updated to reflect these trends. For example, significant events such as the 2002 World Cup finals or the terrorist attack of 11 September 2001 can change web site access pattern dramatically, a change that web linkages often fail to reflect. We have yet to use such human-traversal information for the dynamic, automatic adjustment of web information services.

1.4 Objectives

This thesis undertakes a comprehensive study of the semantic web that provides a common framework that allows data to be shared and reused across applications. Semantic technologies represent meaning using ontologies and provide reasoning through the relationships, rules, logic and conditions represented in those ontologies. The objectives of the thesis can be summarized as follows:

- To explore the advantages of semantic web as compared to plain (HTML) web in several areas such as inadequacies of keyword-based searches and the deficient of automatically constructed directories in the current web.
- To analyze the advantages of XML-SOAP-based system development for the semantic web.
- To examine the semantic web related technologies and the vision of the semantic web, as a new communication and cooperation infrastructure for the web.

1.5 Scope

The scope of this thesis is to study the advantages of the semantic web for data representation in a distributed environment, with the creation of web-based XML-SOAP-based system application.

1.6 Expected Outcome

This system is design based on one of the semantic web initiative that is to make the semantics of web content accessible to machines. The semantic web has been evolving into a web of data separate from the existing HTML web. This work focuses on establishing and exploiting connections between the two webs, especially hyperlink connections from the HTML web pages to the semantic web nodes, so as to enhance both data and document retrieval.

1.7 Significance of the Systems

This is a part of the system with real benefits that can be gained from the whole development of semantic web. The significance of the project can be realized in semantic web development, on a long-term timescale that provides an infrastructure that will enable the evolution of increasingly sophisticated forms of collective intelligence. Ultimately this will result in the web itself becoming more and more intelligent, until one day the entire human species together with all of its software and knowledge will function as something like a single worldwide distributed mind.

1.8 Research Methodology

To achieve objective of this thesis, the research was designed into three major phases. First phase was to conduct a broad view and analysis of existing web and detail discussion of the literature review related to the semantic web architecture. Any materials related to the literature review and system analysis were compiled and filed for future references. From this analysis, only relevant technologies related to semantic web were filtered and embraced. Semantic web related technologies are discussed in detail in Chapter 2. Next phase, from this study, the focus was narrowed to the creation of web-based XML-SOAP-based system application. The chosen software model and system design were discussed in Chapter 3 and Chapter 4. The third phase was focused on the system implementation and conclusion of the research.

1.9 Organization of Thesis

Chapter 1 gives an overview of the thesis, motivation of the study, objectives, scope and the expected outcome.

Chapter 2 contains the related literature review. This chapter discusses various issues regarding the internet and several elements on the semantic web. Overview about the driving principles, architecture, schema and ontology of the semantic web are also included.

Chapter 3 outlines the methodology where it is focused to the software model that been used and the advantages of the software model.

Chapter 4 is devoted to the system analysis and design whereby it is focused the XML-SOAP-based system application. Further details are discussed on the XML technology, SOAP overview and technology, SOAP messages and web services via SOAP.

Chapter 5 provides the details of the system implementation.

Chapter 6 concludes this thesis with conclusion and suggestion for future works.

2.0 LITERATURE REVIEW

The World Wide Web (WWW) is a system of internet servers that support specially formatted documents. The documents are formatted in a language called HTML that supports links to other documents, as well as graphics, audio, and video files. WWW is an information resource with virtually unlimited potential. Recently, researchers have begun to explore the potential of associating web content with explicit meaning in order to create a semantic web. Rather than rely on natural language processing to extract this meaning from existing documents, this approach requires authors to describe documents using a knowledge representation language.

Although data representation can solve many of the webs problems, existing research cannot be directly applied to the semantic web. Unlike most traditional knowledge bases, the web is highly decentralized, changes rapidly, and contains a staggering amount of information. This thesis examines how data representation must change to accommodate these factors. It presents a new method for integrating web data sources based on ontologies, where the sources explicitly commit to one or more autonomously developed ontologies. In addition to specifying the semantics of a set of terms, the ontologies can extend or revise one another. This technique permits automatic integration of sources that commit to ontologies with a common descendant, and when appropriate, of sources that commit to different versions of the same ontology.

2.1 What is the Semantic Web

The web was designed as an information space, with the goal that it should be useful not only for human-human communication, but also that machines would be able to participate and help (Leiner 2003). One of the major obstacles to this has been the fact that most information on the web is designed for human consumption, and even if it was derived from a database with well defined meanings (in at least some terms) for its columns, that the structure of the data is not evident to a robot browsing the web. Leaving aside the artificial intelligence problem of training machines to behave like people, the semantic web approach instead develops languages for expressing information in a machine processable form.

Tim Berners-Lee is the inventor of legendary WWW. Borne in 1955, he is not finish yet for there is a sequel called the semantic web. Its technology is still at infancy stage and naturally, is heavily researched by WWW Consortium (W3C). According to Berners-Lee, “It is a paradigm shift, like the original World Wide Web,” where most people hard to understand what the web was when it was introduced in 1989. “There’s mental leap involved” is what Berners-Lee perceived of public’s reaction to semantic web.

True enough, skeptics view semantic web as rather a far-fetched vision nurtured in academia world when commercially, many major players are racing to reshape the internet, using their own terms and business interest.

2.1.1 Definition of Semantic Web

Semantics is the study of meaning and the semantic web is the old web with a new bunch of acronyms that will let computers understand the meaning of the information on the web, rather than just displaying it to us, as they do at the moment (Berners-Lee 2003).

While according to W3C's definition, semantic web is a mesh of information linked up in such a way that it is easily processable by machines and people on a global scale. It is an evolving collection of data and knowledge, indirectly self-built to allow anyone on the internet to add what they know, to share with others and to find answers to their questions. They could also remove the data and knowledge that they believe are no longer true or maybe perhaps, cease the desire to share it with others.

Another underlying difference from the present web is the people are allowed to remove facts that others put in. This is how the reservoir of information is maintained, apart from in a structured form which is fairly easy for both computers and people to work with. It can be regarded as being an efficient way of representing data on the WWW, or as a globally linked database.

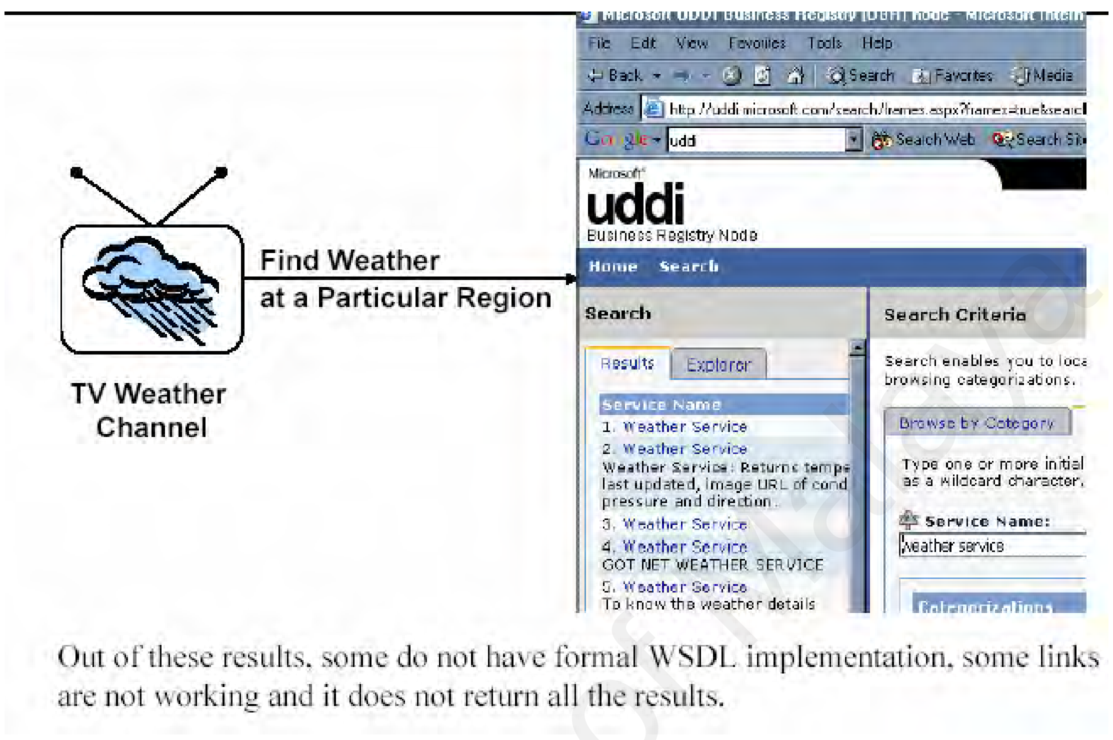
2.1.2 Driving Principles of Semantic Web

One of the main driving forces for semantic web has always been the expression (on the web) of the vast amount of relational database information in a way that can be processed by machines (Manola 2002).

Useful data or information is generally stored in HTML files where the data presentation is personalized, friendly and pleasing to an individual. It is also being stored in a wide variety way that standard is almost non-existent. Though some information will best remain in natural languages, like English, other information is more useful to be stored in a format that is comprehensible for computers to understand (Manola 2002).

In the context of HTML format, it is almost impossible to use the data on a large scale. Unlike relational database, there is no global system or maintenance team that functions to marshal the published data or information in such a way that it could be easily processed by anyone or any machines. For example, there is enormous information about weather all over the world in the internet but none could be consolidated in a fashion to generate a meaningful report for analysis.

Typically in today's web surfing, result of inquiries will be as follow:-



Service Name
1. Weather Service
2. Weather Service
Weather Service: Returns tempe last updated, image URL of cond pressure and direction.
3. Weather Service
4. Weather Service
GOT NET WEATHER SERVICE
5. Weather Service
To know the weather details

Search enables you to loca browsing categorizations.

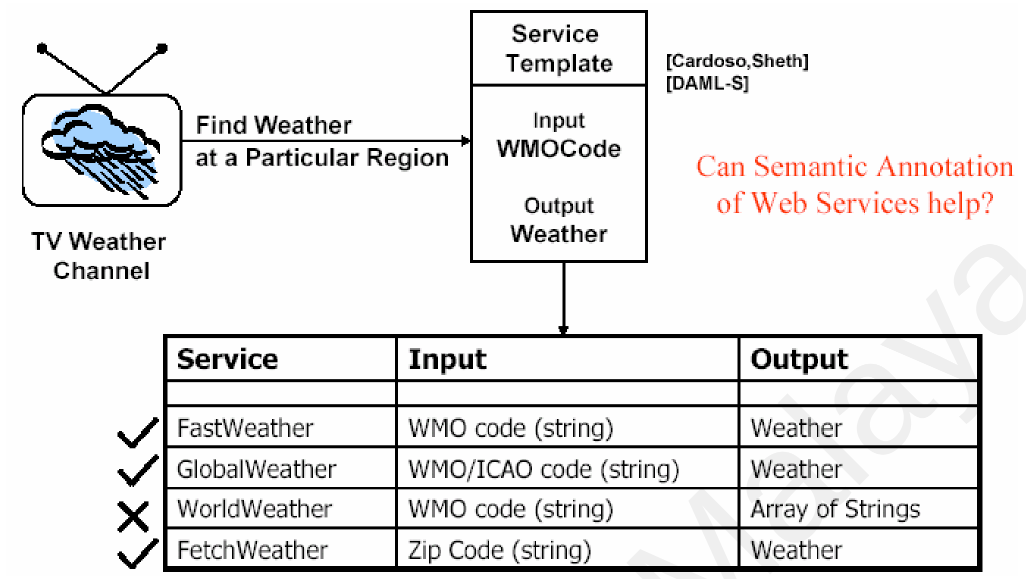
Type one or more initial as a wildcard character,

Out of these results, some do not have formal WSDL implementation, some links are not working and it does not return all the results.

Figure 2.1 : Normal web surfing result of inquiries

(Source: Sheth, Amid 2003. 'Semantic Web Process Lifecycle' University of Georgia, [Online] Available at: http://www.ics.forth.gr/isl/essw2003/talks/seth_essw_semanticwebprocess.htm)

With semantic web concept, it is anticipated that inquiries could be illustrated as follow:-



	Service	Input	Output
✓	FastWeather	WMO code (string)	Weather
✓	GlobalWeather	WMO/ICAO code (string)	Weather
✗	WorldWeather	WMO code (string)	Array of Strings
✓	FetchWeather	Zip Code (string)	Weather

Figure 2.2 : Semantic web surfing result of inquiries

(Source: Sheth, Amid 2003. ‘Semantic Web Process Lifecycle’ University of Georgia, [Online] Available at: http://www.ics.forth.gr/isl/essw2003/talks/seth_essw_semanticwebprocess.htm)

Despite the challenges, there are emerging ‘dot com’ companies who consolidate information from various websites and present it to the convenient of internet surfers, some at a cost. But the maintenance effort is tedious as constant monitoring and enhancement are crucial in view of volatile nature of websites.

Obviously, major re-engineering of the data or information format in the web is required if these data were to be reused in larger scale. The re-engineering should not though, loose the convenience and ease of HTML. Arising from this need, the semantic web was thought and can be seen as a huge engineering solution.

2.1.3 Semantic Web is not an Artificial Intelligence

According to Berners-Lee, semantic web does not imply any shred of artificial intelligence concept though it has some similarities. He argues that, it is just the computer's ability to solve a well-defined set of instruction by filtering well-defined data. It involves a lot of effort from people to ensure that the data and operation are well-defined to ensure the set of instruction are well executed.

Hence, when links are made between the RDF webs, the result will be an expression of a huge amount of information. Semantic web must be able to include all kinds of data to represent the world. RDF at the level with the power of a semantic web will be a complete language, capable of expressing paradox and tautology, and in which it will be possible to phrase questions whose answers would to a machine require a search of the entire web and an unimaginable amount of time to resolve. Therefore, the language itself must be completely expressive.

2.2 Architecture and Data Modeling of Semantic Web

While WWW is presented largely using HTML format, semantic web uses RDF statements where documents and data will be annotated with special codes allowing computers to search and analyze the web automatically. Semantic web is designed to present information in ways that make more sense to computers. Henceforth, web application or software "agents" can analyze the web on behalf of human, making smart inferences that go far beyond the simple linguistic analysis performed by today's search engines like Yahoo and Excite. It would enable correlation about everything that one possesses or desires.

The following picture represents overall relationship of semantic web:-

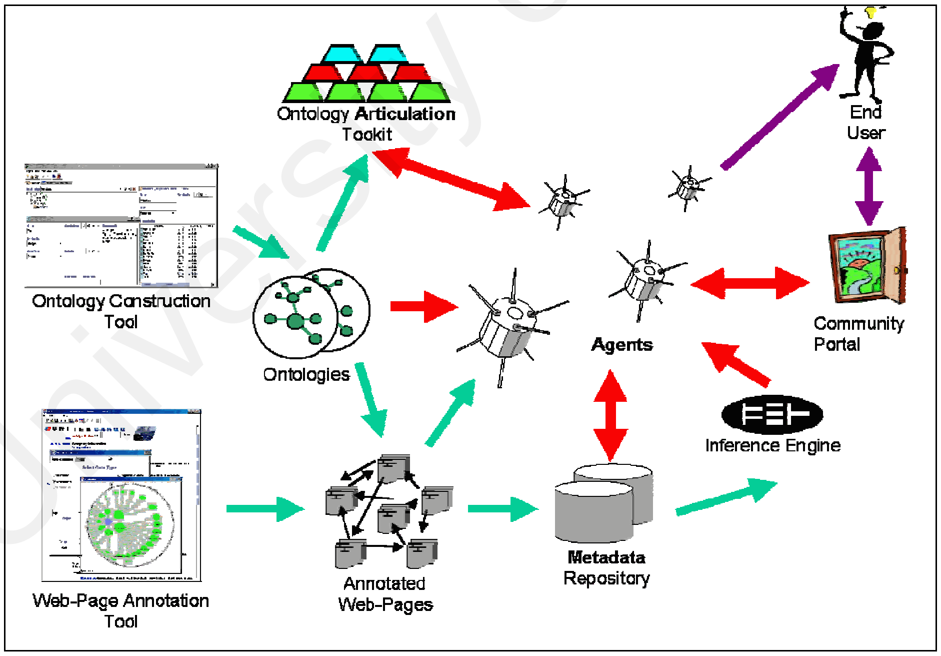


Figure 2.3 : Overall relationship of semantic web

(Source: [Online] Available at: http://www.semanticweb.org/about.html#bigpicture)

The above theory could only be realized if people annotate their web data in advance using URI, RDF and Web Ontology Languages (OWLs) codes. In other words, computers must have access to structured collections of information and sets of inference rules that they can use to conduct automated analysis and reasoning.

The following figure displays overall architecture of semantic web. To date, Logic framework level and upward are still heavily researched and yet to be applied in the real world.

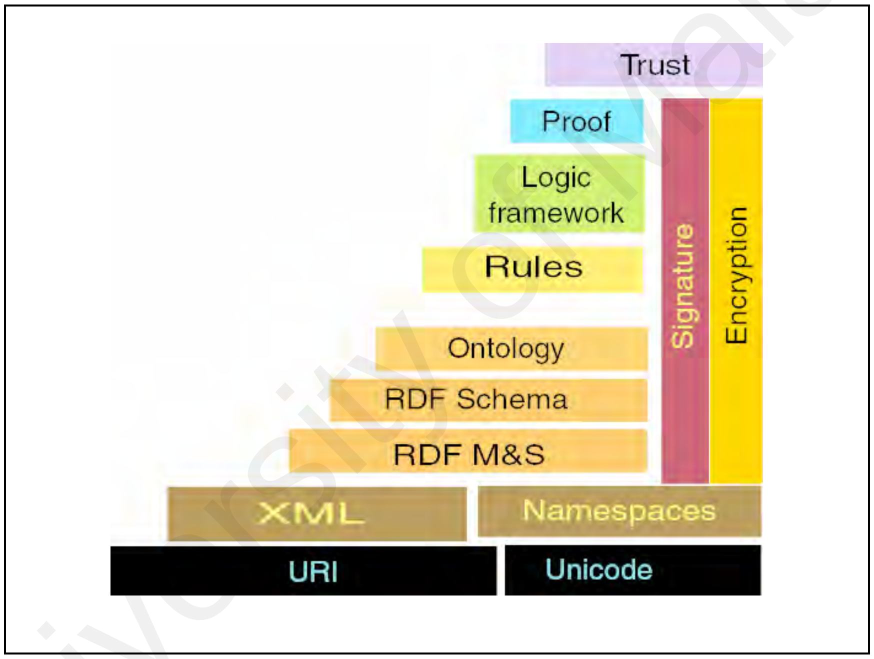


Figure 2.4 : Semantic web architecture

(Source: [Online] Available at : http://www.w3.org/2003/Talks/0922-rsoc-tbl/slide30-0.html)

These structured collections of information and sets of inference rules must ultimately support data integration across application and organizational barriers by ensuring at least the following requirements are met :-

a. Scientific data

   Integration across fields and pointers to experimental conditions, sources, algorithms

b. Electronic Commerce

   Well-defined meaning for documents with catalogs, prices, rates, specifications as data

c. Personal Information Management such as Calendar, Photos, Money

   Semantic web to provide common platform for database and inference, with index rules and ontologies. This field will eventually lead to artificial intelligent.

As mentioned earlier, it is still at infancy stage and involves a lot more complexities especially on logic and rules that govern the syntax, including what privacy rules should govern access to the data. This is where W3C is attempting to set these standards, leading a global collaborative effort among academicians and scientists.

2.3 Current Model of WWW versus Semantic Web

Current model for WWW is islands of disparate XMLs, HTMLs and the users need to manually piece them together to make the information meaningful (Davies 2002). The following diagram reflect the isolation of these documents.

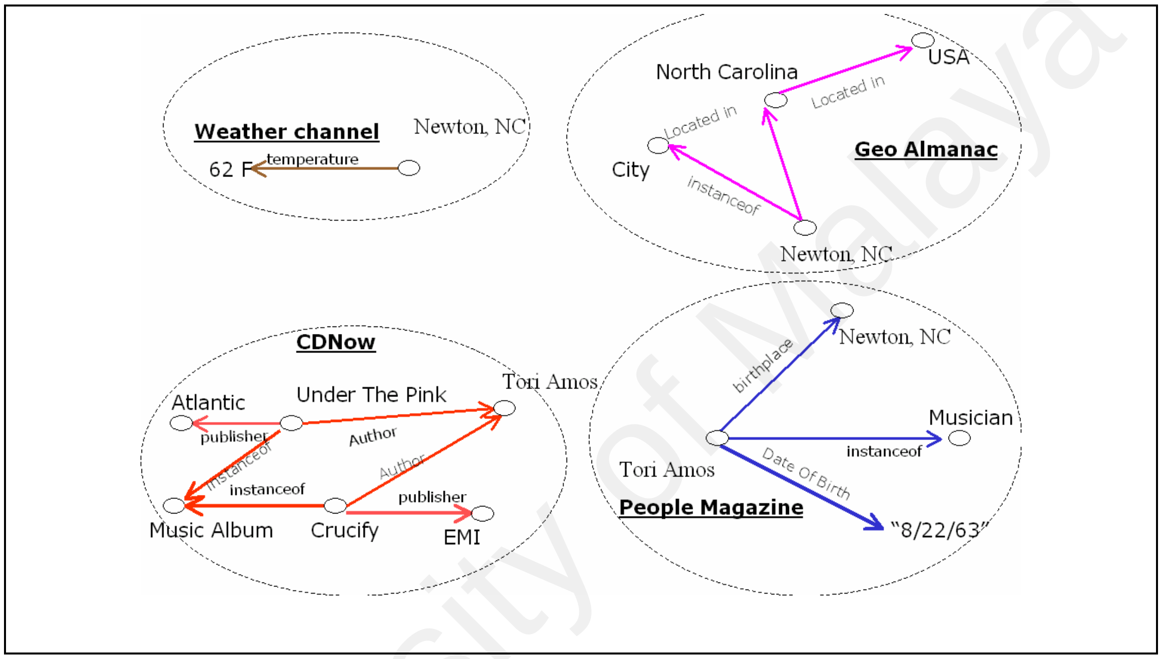


Figure 2.5 : Isolation of data/documents in current model of world wide web

As an alternative to automate the process of piecing the information together, semantic web is seen to be the solution. It will create a coherent data web from disparate chunks to enable client to see a schematically unified view (Davies 2002). The related information effectively create the web as an enormous distributed database; which could be similar to Domain Name Server (DNS.)

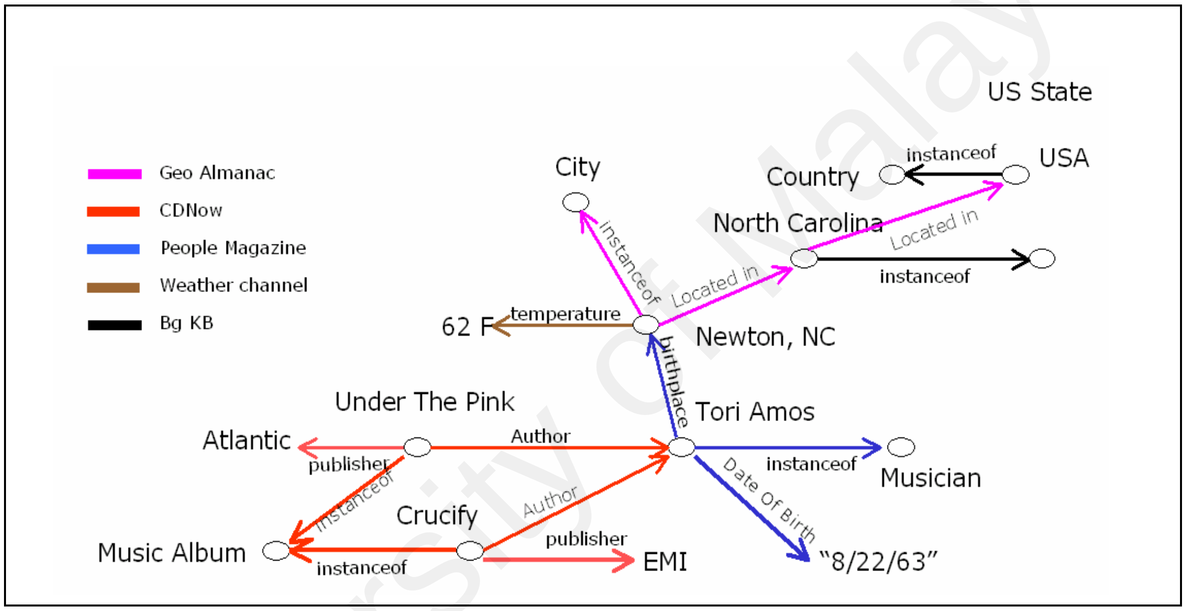


Figure 2.6 : Relation of data/documents in semantic web model

2.4 XML – A New Language for Semantic Web

XML (eXtensible Markup Language) is a markup language for documents containing structured information. Structured information contains both content (words, pictures and others) and some indication of what role that content plays (for example, content in a section heading has a different meaning from content in a footnote, which means something different than content in a figure caption or content in a database table). Almost all documents have some structure. A markup language is a mechanism to identify structures in a document. The XML specification defines a standard way to add markup to documents.

XML is already widely known in the Internet community and is the basis for a rapidly growing number of software development activities (Decker 2000). It is designed for markup in documents of arbitrary structure, as opposed to HTML, which was designed for hypertext documents with fixed structures.

A well-formed XML document creates a balanced tree of nested sets of open and close tags, each of which can include several attribute-value pairs. There is no fixed tag vocabulary or set of allowable combinations, so these can be defined for each application. In XML 1.0, this is done using a DTD (Document Type Definition) to enforce constraints on which tags to use and how they should be nested within a document. A DTD defines a grammar to specify allowable combinations and nestings of tag names, attribute names, and so on. Developments are well under way at W3C to replace DTDs with XML-schema definitions. Although XML schema offers several advantages over DTDs, their role is essentially the same, to define a grammar for XML documents.

XML, is a specification for computer-readable documents. **Markup** means that certain sequences of characters in the document contain information indicating the role of the document's content (Dan 2002). The markup describes the document's data layout and logical structure and makes the information self-describing, in a sense. It takes the form of words between pointy brackets, called *tags* - for example, **<name>** or **<h1>**. In this aspect, XML looks very much like the well-known language HTML.

However, *extensible* indicates an important difference and a main characteristic of XML. XML is actually a *metalanguage*; a mechanism for representing other languages in a standardized way. In other words, XML only provides a data format for structured documents, without specifying an actual vocabulary. This makes XML universally applicable, where we can define customized markup languages for unlimited types of documents. This has already occurred on a massive scale. Besides many proprietary languages, ranging from electronic order forms to application file formats are defined in XML (called *XML applications*). For example, XHTML is a redefinition of HTML 4.0 in XML.

Let's take a more detailed look at XML. The main markup entities in XML are *elements*. They consist normally of an opening tag and a closing tag, for example, **<person>** and **</person>**. Elements might contain other elements or text. If an element has no content, it can be abbreviated as <person/>. Elements should be properly nested, a child element's opening and closing tags must be within its parent's opening and closing tags. Every XML document must have exactly one root element. Elements can carry *attributes* with values, encoded as additional "word = value" pairs inside an element tag - for example, <person name="John">. Here is a piece of XML:

```
<?xml version="1.0"?>
<employees>
  List of persons in company:
  <person name="John">
    <phone>47782</phone>
    On leave for 2001.
  </person>
</employees>
```

XML does not imply a specific interpretation of the data. Of course, on account of the tag's names, the meaning of the previous piece of XML seems obvious to human users, but it is not formally specified. The only legitimate interpretation is that XML code contains named entities with sub entities and values; that is, every XML document forms an ordered, labeled tree. This generality is both XML's strength and its weakness. We can encode all kinds of data structures in an unambiguous syntax, but XML does not specify the data's use and semantics. The parties that use XML for their data exchange must agree beforehand on the vocabulary, its use and its meaning.

XML is foremost a means for defining grammars and because different grammars can be used to describe the same content, XML allows multiple serializations. The information in the final class definition could be expressed in an entirely different form, for example:

```
<class-def>
    <name>branch</name>
    <slot-constraint>
          <name>is-part-of</name>
          <has-value>tree</has-value>
    </slot-constraint>
</class-def>
```

XML is used to serve a range of purposes:

- **Serialization syntax for other markup languages.** For example, the Synchronized Multimedia Integration Language (SMIL) is syntactically just a particular XML DTD; it defines the structure of a SMIL document. The DTD is useful because it facilitates a common understanding of the meaning of the DTD elements and the structure of the DTD.
- **Semantic markup of web pages.** An XML serialization can be used in a web page with an XSL style sheet to render the different elements appropriately.
- **Uniform data-exchange format.** An XML serialization can also be transferred as a data object between two applications.

It is important to note that a DTD specifies only syntactic conventions; any intended semantics are outside the realm of the XML specification.

2.4.1 DTDS and XML Schemas

Such an agreement can be partly specified by DTD and XML Schemas. Although DTDs and XML Schemas do not specify the data meaning, they do specify the names of elements and attributes (the vocabulary) and their use in documents. Both are mechanisms with which we can specify the structure of XML documents. We can then validate specific documents against the structure prescription specified by a DTD or an XML Schema (Bhavani 2002).

DTDs provide only a simple structure prescription, allowed nesting of elements, the elements' possible attributes and the locations where normal text is allowed. For example, a DTD might prescribe that every person element must have a name attribute and may have a child element called phone whose content must be text. A DTDs syntax looks a bit awkward, but it is actually quite simple.

XML Schemas are a proposed successor to DTDs. The XML Schema definition is still a candidate recommendation from the W3C, which means that, although it is considered stable, it might still undergo small revisions. XML Schemas have several advantages over DTDs. First, the XML Schema mechanism provides a richer grammar for prescribing the structure of elements. For example, we can specify the exact number of allowed occurrences of child elements, default values, and can put elements in a *choice* group, which means that exactly one of the elements in that group is allowed at a specific location. Second, it provides data typing. A third advantage is that the XML Schema definition provides inclusion and derivation mechanisms. This lets us reuse common element definitions and adapt existing definitions to new practices.

A final difference from DTDs is that XML Schema prescriptions use XML as their encoding syntax (XML is a metalanguage). This simplifies tool development, because both the structure prescription and the prescribed documents use the same syntax. The XML Schema specification's developers exploited this feature by using an XML Schema document to define the class of XML Schema documents. After all, because an XML Schema prescription is an XML application, it must obey rules for its structure, which can be defined by another XML Schema prescription. However, this recursive definition can be a bit confusing.

2.4.2 RDF (Resource Description Framework) - Represents Data about Data

RDF is the core of semantic web. Similar to XML, RDF comprises statements that are machine-processable where it comprises of simple statements that carry URIs (Decker 2000). Each RDF statement has three parts; a subject, a predicate (a verb or property) and an object (value).

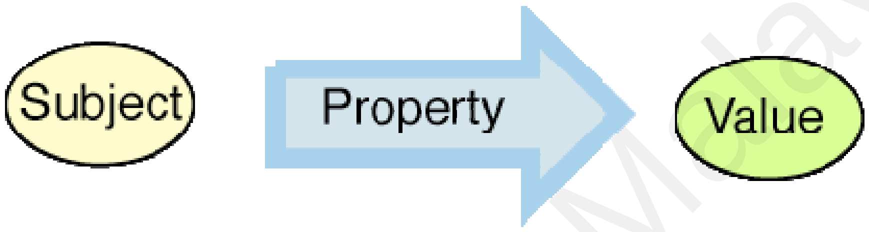


Figure 2.7 : RDF statement

(Source: [Online] Available at: http://www.w3.org/2003/Talks/0922-rsoc-tbl/slide30-0.html)

Having said that, semantic web is generally built on syntaxes which use URIs to represent data, usually in triples based structures; for example, many triples of URI data that can be held in databases or interchanged on the WWW using a set of particular syntaxes developed especially for the task. These syntaxes are called "Resource Description Framework" syntaxes. Example of RDF is illustrated in N-Triples language as follow:-

```
<http://azhartajul.com/> <http://desktop.example.org/terms/reallyLikes>
<http://www.w3.org/People/Berners-Lee/Weaving/>
```

The above statement could be translated as follow:-

Subject	: <http://azhartajul.com/>
Predicate (Verb)	: <http://desktop.example.org/terms/reallyLikes>
Object	: <http://www.w3.org/People/Berners-Lee/Weaving/>

So the RDF statement above says that Azhar Tajul really like "Weaving the Web." The syntax is dynamic that RDF statement can say practically anything and that it doesn't matter who says them. This leads to an important RDF principle, *"anything can say anything about anything."* For instance, one can say contradictory things and it is still acceptable. This is the freedom that the web provides.

The statement above could be rewritten as follow and it still carries the same meaning:-

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:desktop="http://desktop.example.org/terms/">
   <rdf:Description rdf:about="http://azhartajul.com/">
      <desktop:reallyLikes rdf:resource="http://www.w3.org/People/Berners-
Lee/Weaving/"/>
   </rdf:Description>
</rdf:RDF>
```

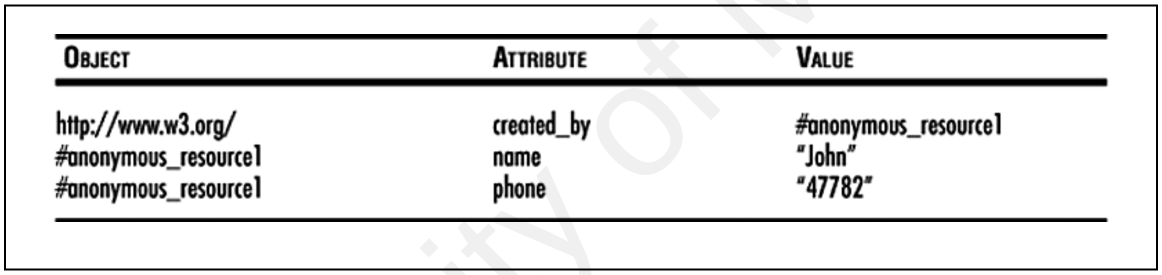
It is the official RDF specification defines an XML representation of RDF, which is a bit more complicated than N-Triple language.

Nonetheless, if one choose to be less ambitious or less visionary, the readily targeted source for RDF is likely to be databases, where thousands most containing interesting machine-processable information exists around the world. For example, companies store part and inventory information in database, computerized phone directories, government store names of people with ids and address. If these databases are published in RDF statements, the possibilities of queries are boundless. It just waits for intelligent programs to fit these data together (Vikram 2002).

XML provides syntax to encode data; the RDF is a mechanism to tell something about data. As its name indicates, it is not a language but a model for representing data about "things on the web." This type of data about data is called *metadata*. The "things" are *resources* in RDF vocabulary.

RDF's basic data model is simple; besides resources, it contains *properties* and *statements*. A property is a specific aspect, characteristic, attribute, or relation that describes a resource. A statement consists of a specific resource with a named property plus that property's value for that resource. This value can be another resource or a *literal* value; free text, basically. Altogether, an RDF description is a list of triples; an object (a resource), an attribute (a property), and a value (a resource or free text). For example, **Table 2.1** shows the three triples necessary to state that a specific web page was created by something with a name "John" and a phone number 47782.

Table 2.1: An RDF description consisting of three triples indicating that a specific web page was created by something with a name John and a phone number 47782



OBJECT	ATTRIBUTE	VALUE
http://www.w3.org/	created_by	#anonymous_resource1
#anonymous_resource1	name	"John"
#anonymous_resource1	phone	"47782"

We can easily depict an RDF model as a directed labeled graph. To do this, we can draw an oval for every resource and an arrow for every property, and we represent literal values as boxes with values. **Figure 2.8** shows such a graph for the triples in **Table 2.1**.

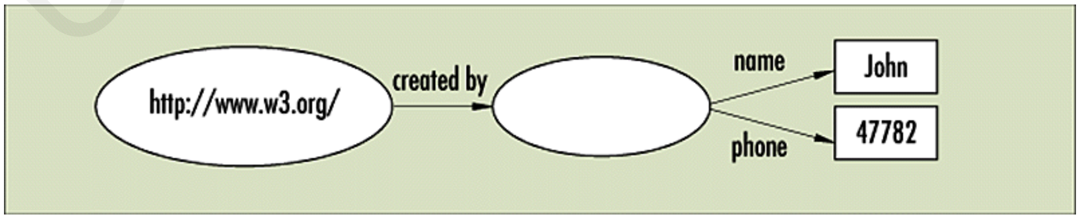


Figure 2.8 : A directed labeled graph for the triples in Table 2.1

These example notations reveal that RDF is ignorant about syntax; it only provides a model for representing metadata. The triple list is one possible representation, as is the labeled graph, and other syntactic representations are possible. Of course, XML would be an obvious candidate for an alternative representation. The specification of the data model includes such an XML-based encoding for RDF (Johan 2001).

As with XML, an RDF model does not define the semantics of any application domain or make assumptions about a particular application domain. It just provides a domain-neutral mechanism to describe metadata. Defining domain-specific properties and their semantics requires additional facilities.

2.5 Semantic Web Service Markup

The automation tasks that have been described are driving the development of the semantic web services markup in the DAML (DARPA Agent Markup Language) family of markup languages, such as:

- Web services, such as Yahoo's driving direction information service or United Airlines' flight booking service;
- User and group constraints and preferences, let's say Ahmad's schedule, that he prefers driving over flying if the driving time to his destination is less than three hours, his use of stock quotes exclusively from the E*Trade web service, and so forth; and
- Agent procedures, which are (partial) compositions of existing web services, designed to perform a particular task and marked up for sharing and reuse by groups of other users. Examples include Ahmad's business travel booking procedure or his friend's stock assessment procedure.

DAML markup provides a declarative representation of web service and user constraint knowledge. A key feature of this markup is the exploitation of ontologies, which DAML+OIL's roots in description logics and frame systems support (Johan 2001).

Some ontology is used to encode the classes and subclasses of concepts and relations pertaining to services and user constraints. Domain-independent web service ontologies are augmented by domain-specific ontologies that inherit concepts from the domain-independent ontologies and that additionally encode concepts that are specific to the individual web service or user. Using ontologies enables the *sharing* of common concepts, the *specialization* of these concepts and vocabulary for *reuse* across multiple applications, the *mapping* of concepts between different ontologies, and the *composition* of new concepts from multiple ontologies. Ontologies support the development of succinct service or user-specific markup by enabling an individual service or user to inherit much of its semantic markup from ontologies, thus requiring only minimal markup at the web site. Most importantly, ontologies can give semantics to markup by constraining or grounding its interpretation. Web services and users need not exploit web service ontologies, but we foresee many domains where communities will want to agree on a standard definition of terminology and encode it in ontology.

2.6 Schemas and Ontology

It is often intuitive to people to relate one entity to another based on given characteristics, behaviors, or even observations but can be very hard to explain to a computer. When properly programmed, however, computers can be very helpful in figuring out which facts follow logically from other facts. Ontologies interweave human understanding of symbols with their machine processability. In a nutshell, ontologies are formal and consensual specifications of conceptualizations, an understanding that can be communicated across people and application systems. It can be expressed in various languages carried by semantic web. Based on the given expression, computers can "understand" the information they are carrying. Semantic web tries to make the meaning so clear that even a dumb computer can understand them. A schema and ontology are ways to describe the meaning and relationships of terms. This description, RDF statements helps computer systems use terms more easily, and decide how to convert between them (Heiner 2005).

At present, no logic language have yet been recommended for semantic web but experimental languages being developed and tested are RDF Schema, DARPA Agent Markup Language with Ontology Inference Layer (DAML+OIL), *swap/log* and *WebOnt*. Conclusively, the facts stored in semantic web will change over time as the true state of world changes. Ontologies must therefore be dynamic living entities whereby it automatically embraces these changes continuously. With each contribution signed and dated, semantic web provides a correct and up-to-date picture.

2.7 Semantic Web Challenges

The current semantic web standards still have to be field tested in everyday e-commerce use, and they face a number of obstacles. For example, Berners-Lee said, vendors and users must develop standard ways of using the technology, so applications will be interoperable.

Gerri Michael Dyer, an Electronic Dissemination Adviser for the US Agency for Health Care Policy and Research, agreed. She anticipated that semantic web technology will initially experience browser compatibility problems. To deal with this, she said, vendors should adhere to open standards. This was a problem when browser makers, particularly Microsoft and Netscape, worked with their own versions of HTML. This meant that if developers did not design multiple versions of their documents for various browsers, some of their documents could not be viewed by all browsers.

Also, Michael-Dyer said that semantic web processes must be transparent to users, so they facilitate not complicate online activities. Another problem will be making sure the many documents in databases will be accessible by semantic web technologies, said Mark Kelley, a Digital Designer for Lighthouse Studios, a web development and design company. Retagging these documents would be a lot of work, he noted. "We need either a clever programming method or a lot of person hours," he said.

2.8 Chapter Summary

In this chapter, we have presented material that discussed the transition of the web technologies. It was also established that the World Wide Web is the biggest repository of information ever created, with growing contents in various languages and fields of knowledge. But, in due course, it is very hard to make sense of this content. Search engines might help to find content containing specific words, but that content might not be exactly what you want. The reason was the search is based on the contents of pages and not the semantic meaning.

Furthermore, data integration applications offer the potential for connecting disparate sources, but they require one-to-one mappings between elements in each different data repository. The semantic web, however, allows a machine to connect to any other machine and exchange and process data efficiently based on built-in, universally available semantic information that describes each resource.

The semantic web augments the current WWW by giving information a well-defined meaning, better enabling computers and people to work in cooperation. This is done by adding machine understandable content to web resources. Such added content is called metadata, whose semantics is provided by referring to ontology, a domain's conceptualization agreed upon by a community. The semantic web relies on the complex interaction of several technologies involving ontologies.

The goal of semantic web research is to transform the web from a linked document repository into a distributed knowledge base and application platform, thus allowing the vast range of available information and services to be more effectively exploited.

Once the semantic web exists, it can provide the ability to tag all content on the web, describe what each piece of information is about and give semantic meaning to the content item. Thus, search engines become more effective than they are now, and users can find the precise information they are hunting.

3.0 METHODOLOGY

In the previous chapter, we have reviewed related literature to identify the semantic web issues and possible solutions. Since there is no “one-size fits all solution” in semantic web issues, we present the software methodology that is systematically examine the semantic web issues and arrive at probable solutions.

3.1 Software Methodology

Software methodology is the practice of using selected process techniques to improve the quality of a software development effort. It is a collection of tools and techniques used to aid system developers in the development of a new system, based on the assumption that a methodical approach to software development results in fewer defects and, therefore, ultimately provides shorter delivery times and better value. These tools and technology differ from discipline to discipline. There are qualitative and quantitative approaches to arriving at a solution.

3.2 Software Model

There are various software models that serve as an abstract representation of the software process. The purpose of a software development process is to produce high quality and timely results without imposing a large overhead on the project. These include:

3.2.1 Waterfall Model

The waterfall involves sequential software development model in which development is seen as flowing steadily downwards through the phases of requirements analysis, design, implementation, testing (validation), integration, and maintenance.

3.2.2 Spiral Model

The spiral model uses combination elements of both design and prototyping-in-stages, in an effort to combine advantages of top-down and bottom-up concepts. Also known as the spiral lifecycle model, it is a systems development method used in information technology. This model of development combines the features of the prototyping model and the waterfall model. The spiral model is intended for large, expensive and complicated projects.

3.2.3 V-model

The V-model is a software development model which can be presumed to be the extension of the waterfall model. Instead of moving down in a linear way, the process steps are bent upwards after the coding phase, to form the typical V shape. The V-Model demonstrates the relationships between each phase of the development life cycle and its associated phase of testing.

3.3 V-model as the Software Methodology

After assessing several software methodologies which are widely used and implemented currently, the V-model has been choosing as the methodology for the web-based XML-SOAP-based system.

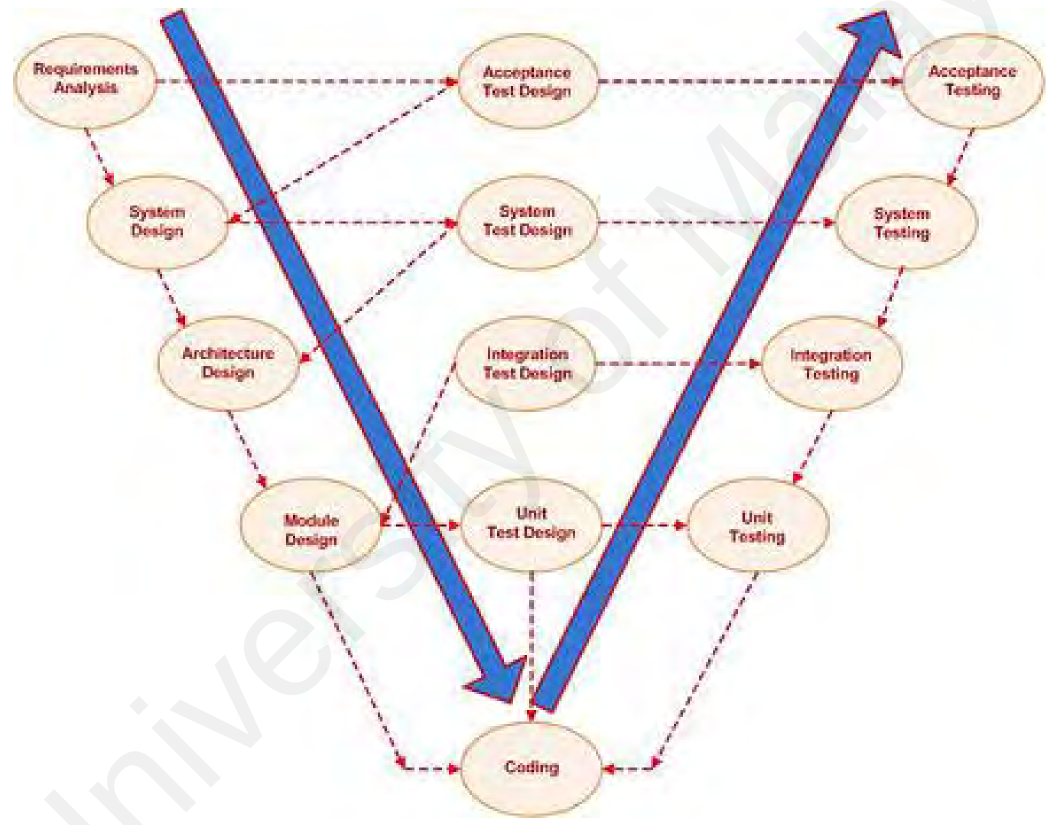


Figure 3.1 : V-model

- **Requirements Analysis** : In this phase, the requirements of the proposed system are collected by analyzing the needs of the user. This phase is concerned about establishing what the ideal system has to perform. However, it does not determine how the software will be designed or built. Usually, the users are interviewed and a document called the user requirements document is generated. The user requirements document will typically describe the system's functional, physical, interface, performance, data, security requirements etc as expected by the user. It is one which the business analysts use to communicate their understanding of the system back to the users. The users carefully review this document as this document would serve as the guideline for the system designers in the system design phase. The user acceptance tests are designed in this phase.
- **System Design** : System engineers analyze and understand the business of the proposed system by studying the user requirements document. They figure out possibilities and techniques by which the user requirements can be implemented. If any of the requirements are not feasible, the user is informed of the issue. A resolution is found and the user requirement document is edited accordingly. The software specification document which serves as a blueprint for the development phase is generated. This document contains the general system organization, menu structures, data structures etc. It may also hold example business scenarios, sample windows, reports for the better understanding. Other technical documentation like entity diagrams, data dictionary will also be produced in this phase. The documents for system testing are prepared in this phase.

- **Architecture Design** : This phase can also be called as high-level design. The baseline in selecting the architecture is that it should realize all which typically consists of the list of modules, brief functionality of each module, their interface relationships, dependencies, database tables, architecture diagrams, technology details etc. The integration testing design is carried out in this phase.
- **Module Design** : This phase can also be called as low-level design. The designed system is broken up in to smaller units or modules and each of them is explained so that the programmer can start coding directly. The low level design document or program specifications will contain a detailed functional logic of the module, in pseudocode - database tables, with all elements, including their type and size - all interface details with complete API references- all dependency issues- error message listings- complete input and outputs for a module. The unit test design is developed in this stage.

3.4 Advantages of V-model

The advantages of V-model are as below:

- Proactive defect tracking i.e. defects are found at early stages even in the development phase before application is tested.
- Avoids the downward flow of the defect.
- Reduces the cost for fixing the defect since defects will be found in early stages.
- It emphasizes the strict process flow to develop a quality product.

4.0 SYSTEM ANALYSIS AND DESIGN

System analysis provides documentation of the system and constraints. The requirements should be precise and serve the purpose in developing the model of system's desired behavior. Thus, for this thesis, our focus is on XML protocols, the use of XML for messaging and remote procedure calls approaches the semantic web from the other end of the spectrum. A key component of XML protocol technology is the description and discovery of web services available via XML protocols such as SOAP (Simple Object Access Protocol), since systems require the ability to conduct electronic transactions with other systems of which they have no prior knowledge. This requirement also has led to the creation of technologies such as WSDL (Web Services Description Language), which describes the characteristics of the interface offered by a web service. On the other hand, SOAP is a protocol for exchanging XML-based messages or data over a computer network, normally using HTTP in a distributed environment. SOAP forms the foundation layer of the web services stack, providing a basic messaging framework that more abstract layers can build on. SOAP can be used to facilitate a Service-Oriented architectural pattern.

There are several different types of messaging patterns in SOAP, but by far the most common is the Remote Procedure Call (RPC) pattern, where one network node *(the client*) sends a request message to another node *(the server*), and the server immediately sends a response message to the client. Indeed, SOAP is the successor of XML RPC.

4.1 SOAP Overview and Name

Originally designed by Dave Winer, Don Box, Bob Atkinson, and Mohsen Al-Ghosein in 1998 with backing from Microsoft (where Atkinson and Al-Ghosein worked at the time) as an object access protocol, the SOAP specification is currently maintained by the XML Protocol Working Group of the W3C.

The name “SOAP” was originally an acronym for *Simple Object Access Protocol*, but the full name was dropped in Version 1.2 of the SOAP specification, because the focus of SOAP shifted from object access to object inter-operability.

4.2 SOAP Transport Methods

Both SMTP and HTTP are valid application layer protocols for SOAP, but HTTP has gained wider acceptance as it works well with today’s internet infrastructure; specifically, SOAP works well with network firewalls. This is a major advantage over other distributed protocols like GIOP/IIOP or DCOM which are normally filtered by firewalls. A key issue under discussion is whether or not HTTP is the right transport given its inherent synchronous nature.

XML was chosen as the standard message format because of its widespread acceptance by major corporations and open source development efforts. Additionally, a wide variety of freely available tools significantly ease the transition to a SOAP-based implementation.

The somewhat lengthy syntax of XML can be both a benefit and a drawback. Its format is easy for humans to read, but can be complex and can have slow processing times. For example, CORBA, GIOP and DCOM use much shorter, binary message formats. On the other hand, hardware appliances are available to accelerate processing of XML messages. Binary XML is also being explored as a means for streamlining the throughput requirements of XML.

4.3 Structure of a SOAP message

A SOAP message is contained in an envelope. Within this envelope are two additional sections; the header and the body of the message. SOAP messages use XML namespaces.

The header contains relevant information about the message. For example, a header can contain the date the message is sent, or authentication information. It is not required, but, if present, must always be included at the top of the envelope.

4.3.1 Example – SOAP messages

Here is an example of how a client might format a SOAP message requesting product information from a fictional warehouse web service. The client needs to know which product corresponds with the ID 827635:

```
<soap:Envelope xmlns:soap=”http://schemas.xmlsoap.org/soap/envelope/”>
  <soap:Body>
    <getProductDetails xmlns=”http://warehouse.example.com/ws”>
      <productID>827635</productID>
    </getProductDetails>
  </soap:Body>
</soap:Envelope>
```

Figure 4.1 : Example of SOAP messages (request from client)

Here is how the warehouse web service might format its reply message with the requested product information:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <getProductDetailsResponse xmlns="http://warehouse.example.com/ws">
      <getProductDetailsResult>
        <productName>Toptimate 3-Piece Set</productName>
        <productID>827635</productID>
        <description>3-Piece luggage set. Black Polyester.</description>
        <price>96.50</price>
        <inStock>true</inStock>
      </getProductDetailsResult>
    </getProductDetailsResponse>
  </soap:Body>
</soap:Envelope>
```

Figure 4.2 : Example of SOAP messages (reply from warehouse)

4.4 Web Services via SOAP

A Web Service is a method that is callable remotely across a network (such as a corporate intranet or the internet itself) that was available via SOAP over HTTP. We can easily imagine similar services made available over the internet. Such web services would differ from traditional content-based internet services. The fundamental difference is this:-

- Content-Based Services serve up web pages (whether static or dynamically generated) for *human consumption*
- Web Services serve up data for *computers*

The entirety of web services available on the internet as a whole is termed the **Service Web.** Let's look at another example. Presumably we are familiar with search engines such as Google which can translate the web content. In this way, we can view an English language version of a web page that was written (or dynamically generated) in Spanish. The translated version is typically generated on the fly by software installed at the search engine's site. Traditionally, if we wanted to set up a new site with similar capabilities, we would have to either write or buy some software to handle the translation and plug it in to your web server somehow. But in the new world of web services it may be possible to offload this work to a site with a dedicated translation web service exploitable via SOAP request.

4.5 System Design

System design is a phase of transforming the system requirements to solutions or system characteristics. Thus, based on the methodology overview and semantic web possible technologies that have been discussed earlier, in order to overview the example of WSDL, and the creation of web services towards the goals and vision for the semantic web, one web-based application, called “Examination Result” had been created using the XAMPP for Windows software. In this application, our focused was on the WSDL and SOAP technologies that applied.

PHP Hypertext Processor and MySQL database have been used for the web pages and the databases. Furthermore, NuSOAP file has been included into the PHP folder, which NuSOAP is a component-based web services toolkit that allows user to send and receive SOAP message over HTTP.

4.5.1 Activity Diagram

First, we have a look at the activity involved. Conceptually, the arrangement for the activity diagram looks like the following:

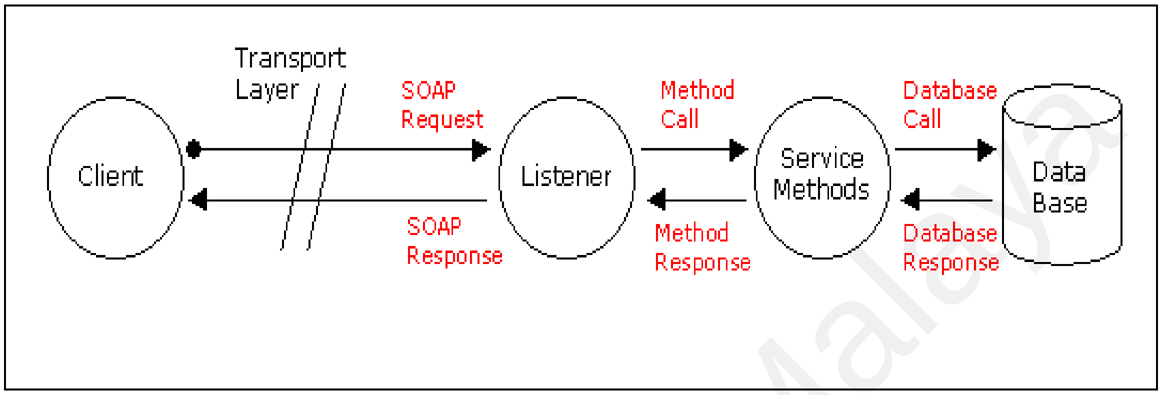
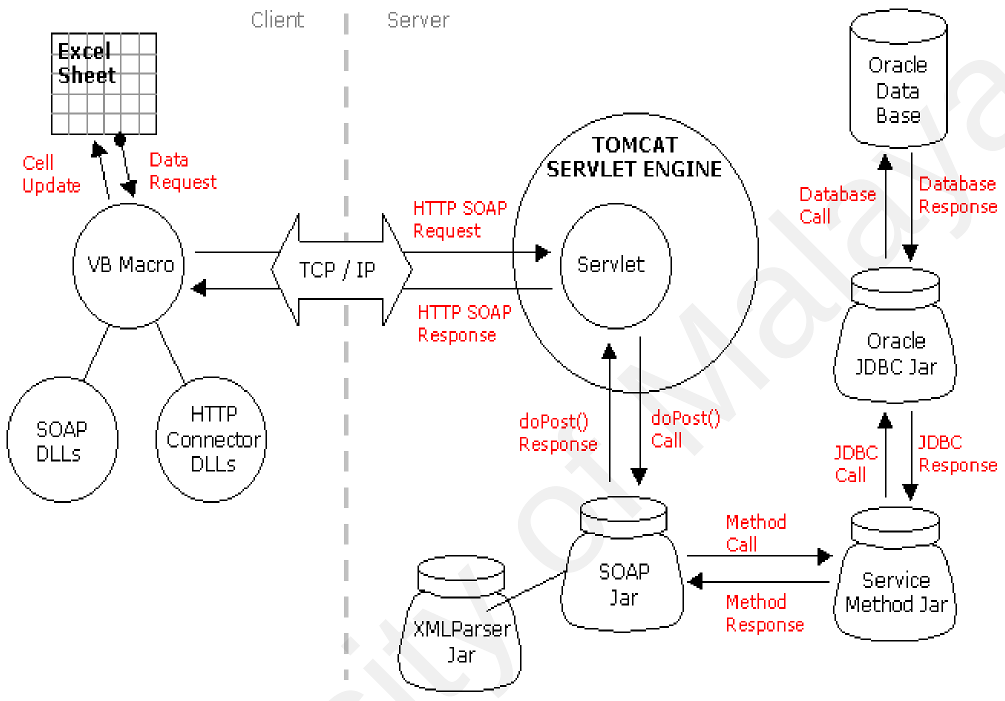


Figure 4.3 : Activity Diagram

While there are many different specific architectures possible for implementing this arrangement, for the purposes of illustration we will summarize one specific possibility.

Let us say the database system is Oracle. The developer writes the service method in Java and connects to the database using an Oracle implementation of JDBC. The listener process is a Java Servlet running within a Servlet Engine such as Tomcat. The servlet has access to some Java classes capable of decoding and encoding SOAP messages (such as Apache SOAP for Java) and is listening for those messages as an HTTP POST. The transport is HTTP over TCP/IP. The client is an excel spreadsheet. It uses a VB Macro which in turn exploits the Microsoft SOAP Toolkit to encode a SOAP request and decode the response received.

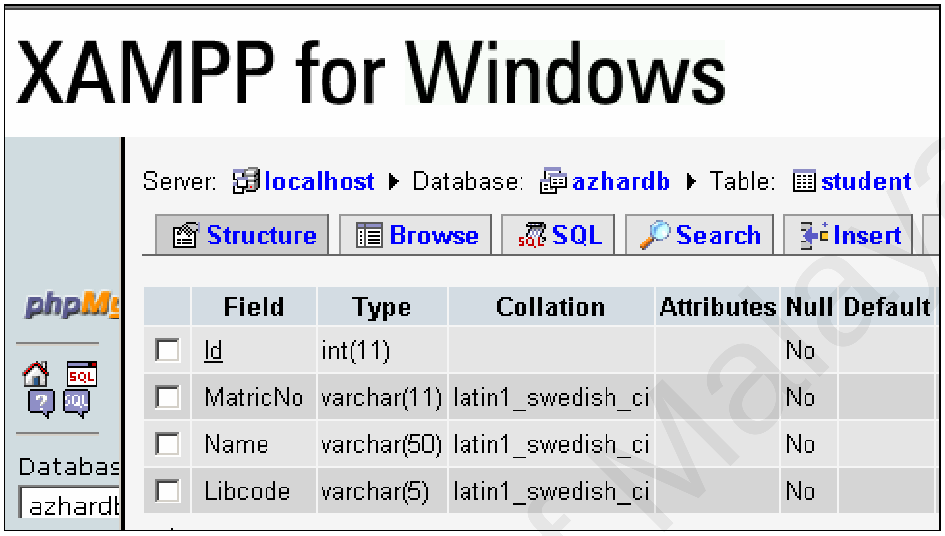
Here is a schematic of that specific implementation looks like:



Note that on the client side the VB Macro relies on both the Microsoft SOAP Toolkit (the SOAP DLLs) and a HTTP Connector interface. Such HTTP Connector DLLs are typically already installed as a part of Internet Explorer. On the server side you will notice that the SOAP package relies on some XML Parser to parse the SOAP messages.

4.5.2 Graphical User Interface (GUI) - “Examination Result” Application

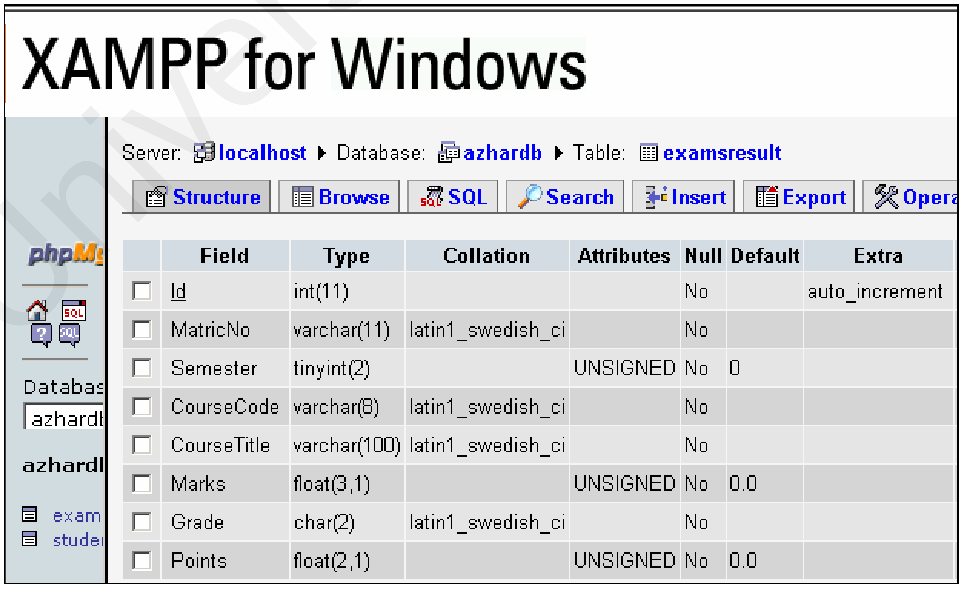
Figure below shows the database structure (student table) in the “Examination Result” application.



	Field	Type	Collation	Attributes	Null	Default
☐	Id	int(11)			No	
☐	MatricNo	varchar(11)	latin1_swedish_ci		No	
☐	Name	varchar(50)	latin1_swedish_ci		No	
☐	Libcode	varchar(5)	latin1_swedish_ci		No	

Figure 4.4 : Database structure (student table) in the “Examination Result” application.

Figure below shows the database structure (examresult table) in the “Examination Result” application.



	Field	Type	Collation	Attributes	Null	Default	Extra
☐	Id	int(11)			No		auto_increment
☐	MatricNo	varchar(11)	latin1_swedish_ci		No		
☐	Semester	tinyint(2)		UNSIGNED	No	0	
☐	CourseCode	varchar(8)	latin1_swedish_ci		No		
☐	CourseTitle	varchar(100)	latin1_swedish_ci		No		
☐	Marks	float(3,1)		UNSIGNED	No	0.0	
☐	Grade	char(2)	latin1_swedish_ci		No		
☐	Points	float(2,1)		UNSIGNED	No	0.0	

Figure 4.5 : Database structure (student table) in the “Examination Result” application.

Next, the below figure shows the main page of the “Examination Result” web-based application that has been created.

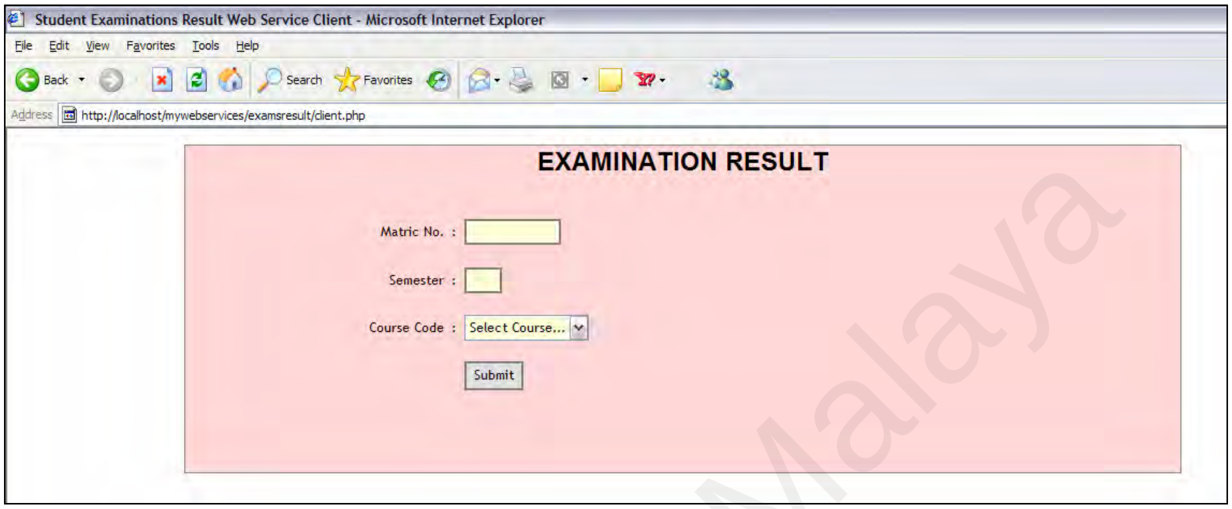
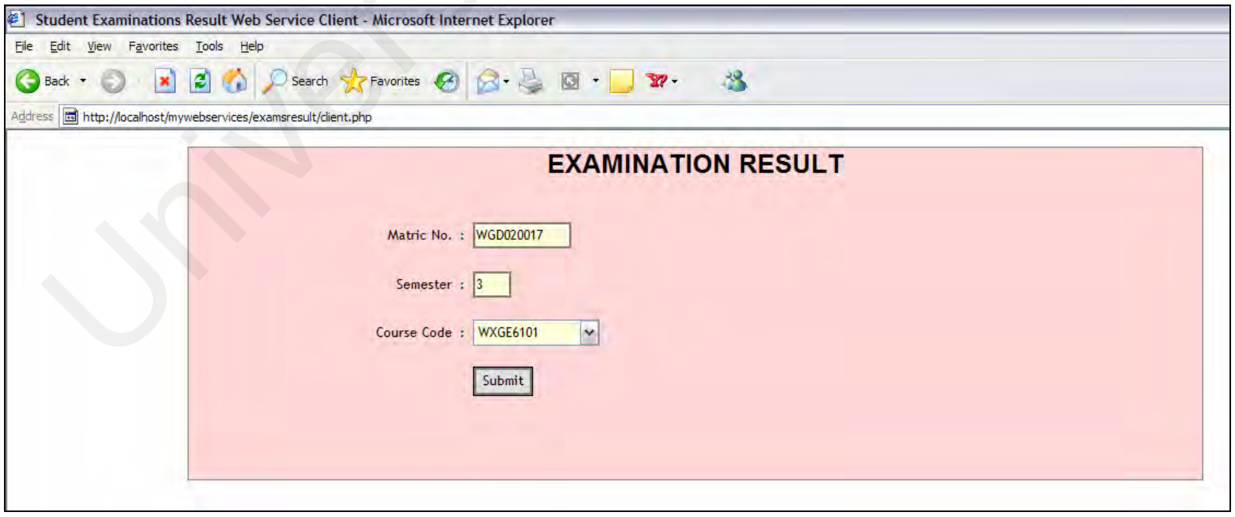

Student Examinations Result Web Service Client - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/mywebservices/examsresult/client.php
EXAMINATION RESULT
Matric No. :
Semester :
Course Code : Select Course...
Submit

Figure 4.6 : “Examination Result” web-based application main page

On this page, user will enter all data that required (Matric No, Semester and Course Code) and click on the “Submit” button to retrieve the result.

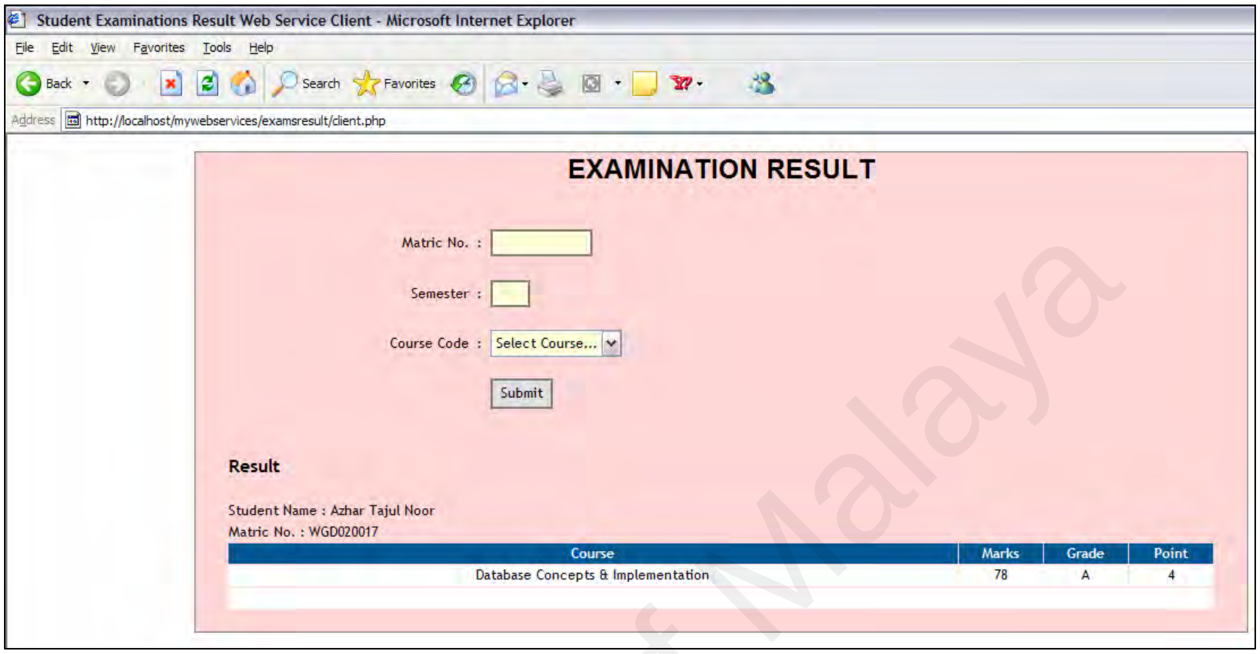

Student Examinations Result Web Service Client - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/mywebservices/examsresult/client.php
EXAMINATION RESULT
Submit

Matric No. : WGD020017

Semester : 3

Course Code : WXGE6101

Next, the result for specific subject that has been entered by user will be displayed.


Student Examinations Result Web Service Client - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/mywebservices/examsresult/client.php

EXAMINATION RESULT

Matric No. :

Semester :

Course Code : Select Course...

Submit

Result

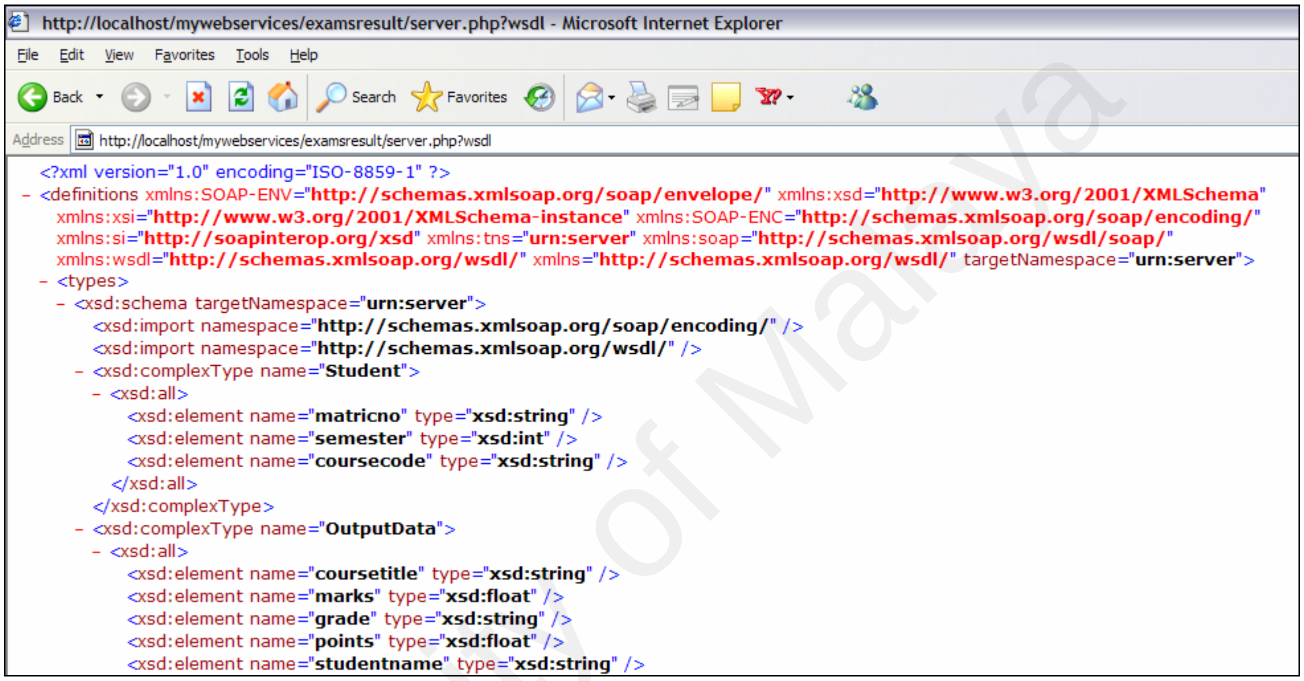
Student Name : Azhar Tajul Noor
Matric No. : WGD020017

Course	Marks	Grade	Point
Database Concepts & Implementation	78	A	4

Figure 4.7 : "Examination Result" web-based application result page

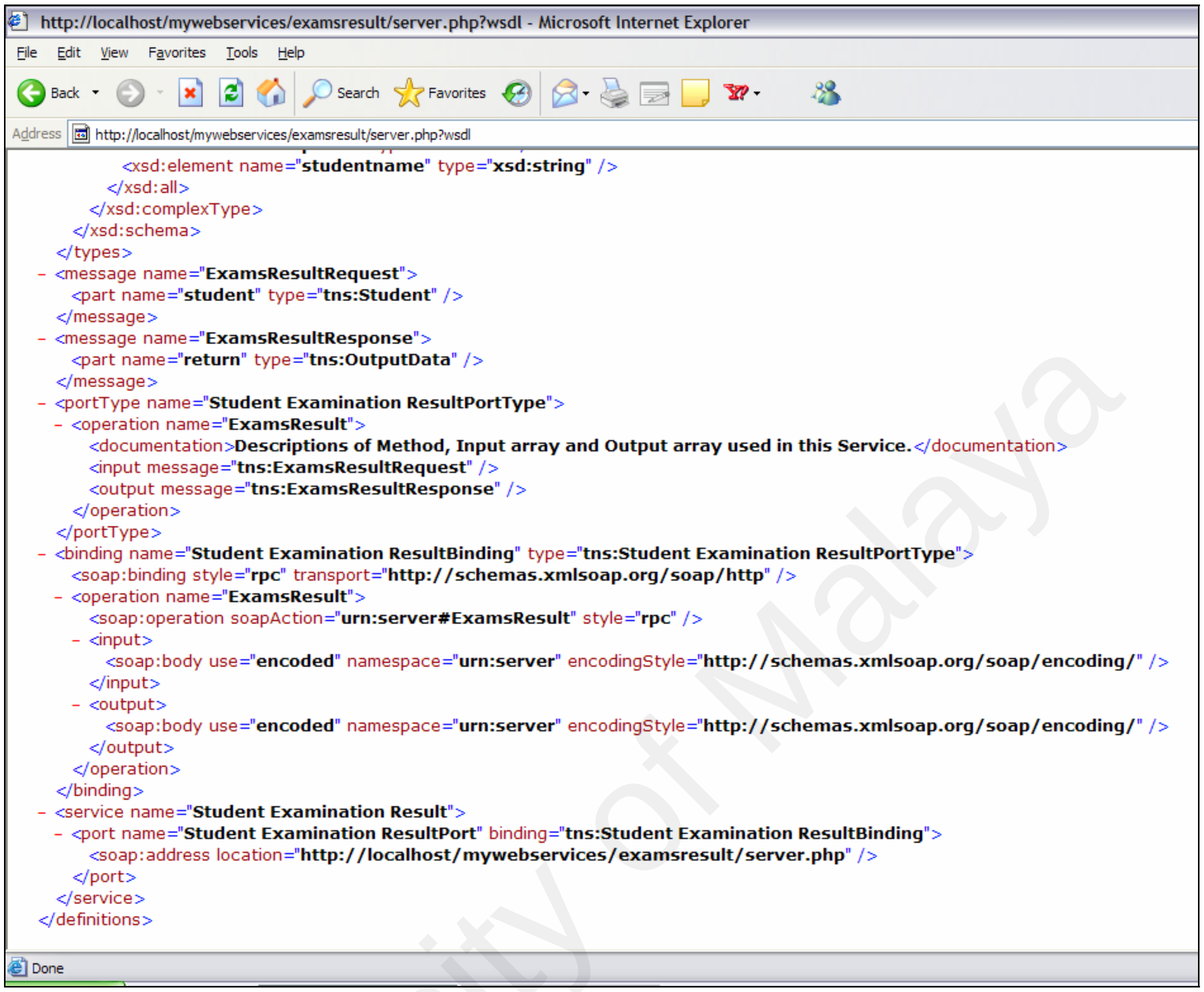
As our focused was on the WSDL and SOAP technologies that applied, we will look into the SOAP messages that have been created when we retrieve the data thru the web page.

Figure below shows the examples of SOAP messages:


http://localhost/mywebservices/examsresult/server.php?wsdl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/mywebservices/examsresult/server.php?wsdl

```
  <?xml version="1.0" encoding="ISO-8859-1" ?>
- <definitions xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:si="http://soapinterop.org/xsd" xmlns:tns="urn:server" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns="http://schemas.xmlsoap.org/wsdl/" targetNamespace="urn:server">
  - <types>
    - <xsd:schema targetNamespace="urn:server">
        <xsd:import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
        <xsd:import namespace="http://schemas.xmlsoap.org/wsdl/" />
      - <xsd:complexType name="Student">
        - <xsd:all>
            <xsd:element name="matricno" type="xsd:string" />
            <xsd:element name="semester" type="xsd:int" />
            <xsd:element name="coursecode" type="xsd:string" />
          </xsd:all>
        </xsd:complexType>
      - <xsd:complexType name="OutputData">
        - <xsd:all>
            <xsd:element name="coursetitle" type="xsd:string" />
            <xsd:element name="marks" type="xsd:float" />
            <xsd:element name="grade" type="xsd:string" />
            <xsd:element name="points" type="xsd:float" />
            <xsd:element name="studentname" type="xsd:string" />
```

Figure 4.8 : Example of the SOAP messages created from the “Examination Result” web-based application result page


http://localhost/mywebservices/examsresult/server.php?wsdl - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/mywebservices/examsresult/server.php?wsdl
Done

```
        <xsd:element name="studentname" type="xsd:string" />
      </xsd:all>
    </xsd:complexType>
  </xsd:schema>
</types>
- <message name="ExamsResultRequest">
    <part name="student" type="tns:Student" />
  </message>
- <message name="ExamsResultResponse">
    <part name="return" type="tns:OutputData" />
  </message>
- <portType name="Student Examination ResultPortType">
  - <operation name="ExamsResult">
      <documentation>Descriptions of Method, Input array and Output array used in this Service.</documentation>
      <input message="tns:ExamsResultRequest" />
      <output message="tns:ExamsResultResponse" />
    </operation>
  </portType>
- <binding name="Student Examination ResultBinding" type="tns:Student Examination ResultPortType">
    <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http" />
  - <operation name="ExamsResult">
      <soap:operation soapAction="urn:server#ExamsResult" style="rpc" />
    - <input>
        <soap:body use="encoded" namespace="urn:server" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
      </input>
    - <output>
        <soap:body use="encoded" namespace="urn:server" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
      </output>
    </operation>
  </binding>
- <service name="Student Examination Result">
  - <port name="Student Examination ResultPort" binding="tns:Student Examination ResultBinding">
      <soap:address location="http://localhost/mywebservices/examsresult/server.php" />
    </port>
  </service>
</definitions>
```

Figure 4.9 : Example of the SOAP messages created from the "Examination Result" web-based application result page

In the above SOAP messages examples, it shows that SOAP is responsible for encoding messages in a common XML format so that messages can be understood at either end. Currently, this includes XML-RPC and SOAP. For example, the client program bundles up two values to be added into a SOAP message, which is sent to the web service by sending it as the body of an HTTP POST **request**. The server unpacks the SOAP request that the application can understand and executes add operation. Next, the server packages up that result of summation as **response** into another SOAP message, which it sends back to the client program in response to its HTTP request. The client program unpacks the SOAP message to obtain the results of the summation. So, SOAP = XML + HTTP.

Below figure shows the NuSOAP **request** page for “Examination Result” web-based application:

```
POST /MyWebServices/examsresult/server.php HTTP/1.0
Host: localhost
User-Agent: NuSOAP/0.7.1 (1.91)
Content-Type: text/xml; charset=ISO-8859-1
SOAPAction: "urn:server#ExamsResult"
Content-Length: 706

<?xml version="1.0" encoding="ISO-8859-1"?><SOAP-ENV:Envelope SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:si="http://soapinterop.org/xsd" xmlns:tns="urn:server"><SOAP-
ENV:Body><tns:ExamsResult xmlns:tns="urn:server"><student
xsi:type="tns:Student"><matricno
xsi:type="xsd:string">WGD020017</matricno><semester
xsi:type="xsd:int">3</semester><coursecode
xsi:type="xsd:string">WXGE6101</coursecode></student></tns:ExamsResult>
</SOAP-ENV:Body></SOAP-ENV:Envelope>
```

Figure 4.10 : NuSOAP **request** page for “Examination Result” web-based application

Below figure shows the NuSOAP **response** page for “Examination Result” web-based application:

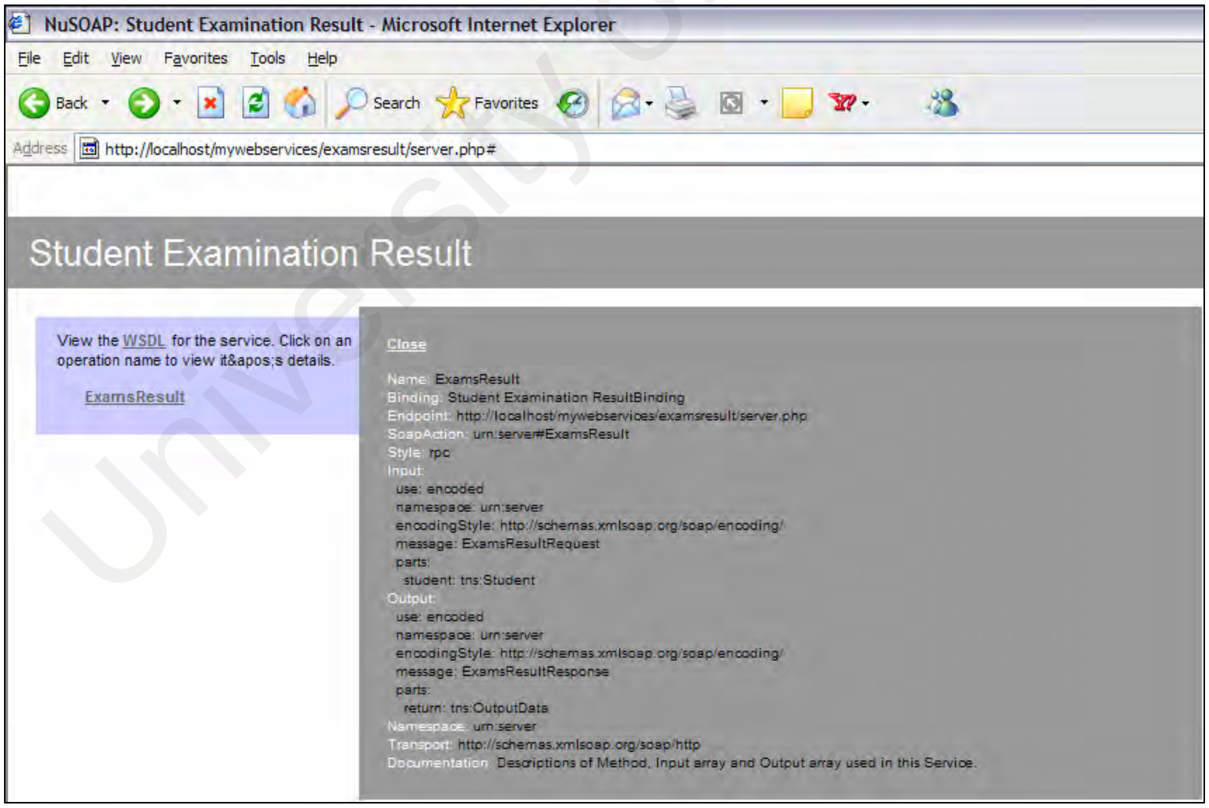
```
HTTP/1.1 200 OK
Date: Tue, 12 Apr 2005 16:46:54 GMT
Server: Apache/2.0.53 (Win32) mod_ssl/2.0.53 OpenSSL/0.9.7f PHP/5.0.4
X-Powered-By: PHP/5.0.4
X-SOAP-Server: NuSOAP/0.7.1 (1.91)
Content-Length: 846
Connection: close
Content-Type: text/xml; charset=ISO-8859-1

<?xml version="1.0" encoding="ISO-8859-1"?><SOAP-ENV:Envelope SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:si="http://soapinterop.org/xsd" xmlns:tns="urn:server"><SOAP-
ENV:Body><ns1:ExamsResultResponse xmlns:ns1="urn:server"><return
xsi:type="tns:OutputData"><coursetitle xsi:type="xsd:string">Database
Concepts &amp; Implementation</coursetitle><marks
xsi:type="xsd:float">78.0</marks><grade
xsi:type="xsd:string">A</grade><points
xsi:type="xsd:float">4.0</points><studentname
xsi:type="xsd:string">Azhar Tajul
Noor</studentname></return></ns1:ExamsResultResponse></SOAP-
ENV:Body></SOAP-ENV:Envelope>
```

Figure 4.11 : NuSOAP **response** page for “Examination Result” web-based application

On the other hand, WSDL is responsible for describing the public interface to a specific web service, which is an XML-based language that describes the various functions that a web service is capable of. Currently, service description is handled via the WSDL to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format. Other systems interact with the web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other web-related standards.

Figure below shows the example of WSDL page that has been created for "Examination Result" web-based application:-


NuSOAP: Student Examination Result - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address

http://localhost/mywebservices/examsresult/server.php#

Student Examination Result

View the WSDL for the service. Click on an operation name to view it&apos;s details.

ExamsResult

Close

```
Name: ExamsResult
Binding: Student Examination ResultBinding
Endpoint: http://localhost/mywebservices/examsresult/server.php
SoapAction: urn:server#ExamsResult
Style: rpc
Input:
  use: encoded
  namespace: urn:server
  encodingStyle: http://schemas.xmlsoap.org/soap/encoding/
  message: ExamsResultRequest
  parts:
   student: tns:Student
Output:
  use: encoded
  namespace: urn:server
  encodingStyle: http://schemas.xmlsoap.org/soap/encoding/
  message: ExamsResultResponse
  parts:
   return: tns:OutputData
Namespace: urn:server
Transport: http://schemas.xmlsoap.org/soap/http
Documentation: Descriptions of Method, Input array and Output array used in this Service.
```

5.0 SYSTEM IMPLEMENTATION

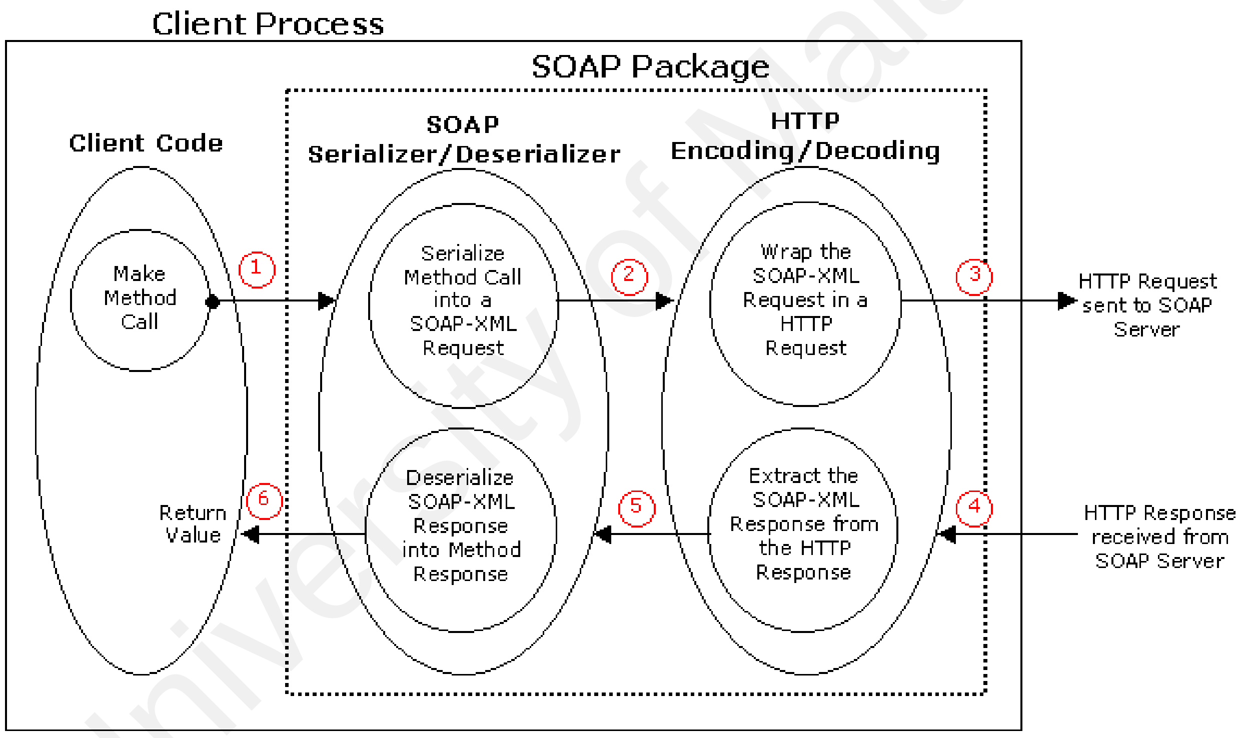
Now, based on the Examination Result web-based application that has been created, we will discuss further the system implementation in a typical SOAP-based system, what they do and how they interact.

Let's look at the **Client-Side** first. As we have seen, all our dialogues with the server running the web service are done via SOAP. Remember, SOAP does not specify a transport but HTTP is common. This example presumes the use of HTTP (but we could just as easily use SMTP). As such, our messages to the server will be SOAP-XML requests wrapped in HTTP requests. Likewise the responses from the server will be HTTP responses that enclose SOAP-XML responses.

Now, we as client-side developers do not want to have to worry about all the details of SOAP serialization and HTTP encoding, so we employ a SOAP package to do this for us. This is typically a library that we link into our own client code. We would then invoke services simply by invoking the appropriate method in the SOAP package (typically specifying the service URL, service name and all required parameters). The first job of the SOAP package is to *serialize* this service invocation into a SOAP request. It then needs to *encode* that message in a HTTP request and send it to the specified URL.

Thus, we rely on the same SOAP package to do the reverse of what was done at the request stage, for example we rely on it to *decode* the HTTP message and extract the SOAP message, then *desterilize* the SOAP message and obtain the return value of the method call. The return value found is then passed as the return value to the original method invocation by the client code.

Here is a graphical depiction of the process:-


Client Process
SOAP Package
Client Code
SOAP Serializer/Deserializer
HTTP Encoding/Decoding
Make Method Call
1
Serialize Method Call into a SOAP-XML Request
2
Wrap the SOAP-XML Request in a HTTP Request
3
HTTP Request sent to SOAP Server
Return Value
6
Deserialize SOAP-XML Response into Method Response
5
Extract the SOAP-XML Response from the HTTP Response
4
HTTP Response received from SOAP Server

On the **Server-Side**, it is slightly more complex as we need to have a listener process. We also need an implementation of the service itself. But aside from that we rely on a SOAP package in a similar way as on the client side.

The listener process is often implemented using a servlet running as a web application on an application server (as is the case when we use Apache SOAP on the server side). The application server will be set up to pass all requests for a certain URL (the URL for the SOAP service) to a particular servlet (let's call it the SOAP servlet). The job of the SOAP servlet is to extract the XML-SOAP message from the HTTP request, deserialize it (thereby separating out the method name and the supplied parameters), and invoke the service method accordingly. The result of the method is then serialized, HTTP-encoded and sent back to the requester.

6.0 EVALUATION AND CONCLUSION

The essential aim of this thesis is to examine the advantages of semantic web compared to plain (HTML) web. Thus, as we have developed the “Examination Result” application by using the XML-SOAP-based system towards the realization of the semantic web, it shows that XML-SOAP-based system has solved several cases of interest. First, XML-SOAP-based system has paved the road to the web by adding some metadata in the form of human-readable tags that describe data. Next, XML-SOAP-based documents also include information about the author of a web page, relevant keywords for search engine optimization, and the software tools used to create the file, that helps the tags to be understandable not just to humans, but to machines as well.

The semantic web, on the other hand, is about having data as well as documents on the web so that machines can process, transform, assemble, and even act on the data in useful ways. Furthermore, in the semantic web, data itself becomes part of the web and is able to be processed independently of application, platform, or domain. This is in contrast to the plain (HTML) web as we know it today, which contains virtually boundless information in the form of documents. We can use computers to search for these documents, but they still have to be read and interpreted by humans before any useful information can be extrapolated. So, the vision of the semantic web is a “web of data” that not only harnesses the seemingly endless amount of data on the WWW, but also connects that information with data in relational databases and other non-interoperable information repositories.

Thus, for future research, it is recommended that as the semantic web is to be built on top of the web, many of its characteristics are there as a base and should be continued. Right now, the semantic web kind of search capability is impossible because web search engines require that users guess the right keywords to find what they seek. However, several maturing technologies are considered the most likely keys to fulfilling the goals of the semantic web project. These technologies such as DAML and OWL will help search engines discern whether two web sites have the same content even if they are described using different terminology or metalanguage, already tried and tested in research labs, will help make the semantic web a reality as the web provides the ecology in which the semantic web must thrive, not destroy.

REFERENCES

Berners-Lee, T. Hendler, J., Lassila, O. 2003, *How the Semantic Work.* [Online] Available at: http://www.w3.org/2002/03/semweb/

Berners-Lee, *What the Semantic Web can represent,* [Online] Available at: http://www.w3.org/DesignIssues/RDFnot.html

Berners-Lee, 2003, *WWW Past & Future,* [Online] Available at: http://www.w3.org/2003/Talks/0922-rsoc-tbl/Overview.html

Clabby, Joe 2002, *Web Services Explained : Solutions and Applications for the Real World,* Prentice Hall, Upper Saddle River, New Jersey.

Davies, John & Fensel, Dieter (eds) 2002, *Towards the Semantic Web: Ontology-driven Knowledge Management***,** John Wiley & Sons, Inc. New York.

Decker, S. & Melnik, S. 2000, *The Semantic Web : the roles of XML and RDF,* Stanford University, California.

Dumbill, Edd, *The Semantic Web,* [Online] Available at: http://writetheweb.com/Members/edd/old/87/view

Dunham, Margaret 2002, *Data Mining : Introductory and Advanced Topics*, Prentice Hall, New Jersey.

Dustin, Elfriede 2002, *Quality Web Systems : Performance, Security and Usability*, Addison Wesley, Boston.

Goldfarb, Charles F. 2000, *The XML Handbook*, Prentice Hall, Upper Saddle River, New Jersey.

Hjelm, Johan 2001, *Creating the Semantic Web with RDF*, John Wiley & Sons, Inc. New York.

Fensel, D, *Intelligent Web Services*, [Online] Available at: http://informatik.uibk.ac.at/users/c70385/ftp/slides/finland.ppt

Jenkins, C., Jackson, M., Burden, P., and Wallis, J. 2003, *Automatic RDF Metadata Generation for Resource Discovery*. [Online] Available at: http://people.cs.uct.ac.za/~hbrown/CAKMS/articles/Automa.pdf

Klein, M. 2001, *Tutorial : The Semantic Web - XML, RDF, and Relatives*, [Online] Available at: http://www.swi.psy.uva.nl/Capita-AI/2002/papers/klein.pdf

Lagoze, C. 1996, *The Warwick Framework - A container architecture for diverse sets of metadata*, [Online] Available at: http://www.dlib.org/dlib/july96/lagoze/07lagoze.html

Lassila, O. & Swick, R. R. 1999, *Resource Description Framework (RDF) Model and Syntax Specification*, [Online] Available at: http://www.w3.org/TR/PR-rdf-syntax

Leiner, B. M., Cerf, V. G., Clark, D. D., Kahn, R. E., Kleinrock, L., Lynch, D. C., Postel, J., Roberts, L. G., &Wolff, S. 2003, *A Brief History of the Internet*, [Online] Available at: http://www.isoc.org/internet/history/brief.shtml

Livingston, Dan 2002, *Advanced SOAP for Web Professionals*, Prentice Hall, New Jersey.

Manola, Frank 2002, *The Semantic Web and the Role of Information Systems Research*, Onto Web Invitational Workshop on DB-IS Research for Semantic Web and Enterprises, April 3-5, 2002 Bedford, Massachusetts.

Pressman, R. S. 2001, *Software Engineering - A Practitioner's Approach*, McGraw-Hill.

Seely, Scott 2002, *SOAP : Cross Platform Web Service Development Using XML*, Prentice Hall PTR, Upper Saddle River, New Jersey

Sheth, Amid 2003, *Semantic Web Process Lifecycle*, [Online] Available at:
http://lsdis.cs.uga.edu/lib/presentations/WWW2003-ESSW-invitedTalk-Sheth.pdf

Staab, Steffen & Stuckenschmidt, Heiner 2006, *Semantic Web and Peer-to-peer : Decentralized Management and Exchange of Knowledge and Information*, Springer, New York

Stuckenschmidt, Heiner & Harmelen, Frank van. 2005, *Information Sharing on the Semantic Web*, Springer, New York

Thuraisingham, Bhavani 2002, *XML Databases and the Semantic Web*, CRC Press, Boca Raton.

Vikram, Vaswani 2002, *XML and PHP*, New Riders, Indiana.