

Integrating Semantic Web with Knowledge Management Agent

By

Mohd Shahrul Anuar B. Ishak

Dissertation submitted in partial fulfillment of
the requirements for the
Bachelor of Technology (Hons)
(Information System)

JULY 2005

Universiti Teknologi PETRONAS

Bandar Seri Iskandar

31750 Tronoh

Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Integrating Semantic Web With Knowledge Management Agent

by

Mohd Shahrul Anuar B. Ishak

A project dissertation submitted to the
Information System Program
Universiti Teknologi PETRONAS
in partial fulfillment of the requirement for the
BACHELOR OF TECHNOLOGY (Hons)
(INFORMATION SYSTEM)

Approved by,



(MRS. MAZEYANTI MOHD ARIFFIN)

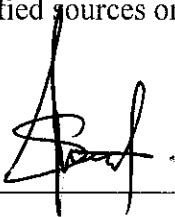
UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

July 2005

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.



MOHD SHAHRUL ANUAR ISHAK

ABSTRACT

This paper is about the fundamental of a Semantic Web Service system. This system is always related to a search engine. By integrating semantic web with a knowledge representation agent, this will help user to reduce the searching time, as the agent will classify the output of their search using this web service. Besides, by having such system it also can ease the user in form of interacting with the data and to appreciate the data management and knowledge management. Having a Semantic web is quite important as Semantic web will be a platform, where the knowledge management (KM), will applied and perform their task in doing the classification of the data. Semantic web need to be constructed in user-friendly environment in which user can use the Semantic web as a channel to transfer knowledge from user to the system. The construction of semantic web require a certain framework and language as a tools, this will add value to the database as it will query the related web link in a particular manner. Beside, semantic web will also include URI as its framework, and one of the URI been used is RDF (Resource Description Framework) in which Once information is in RDF form, it becomes easy to process it, since RDF is a generic format, which already has many parsers. In addition, this will satisfied the last objective of building the semantic web. The need of semantic web service are based on the problem faced by user as when they use any search engine to obtain some information, the output of the search will produce the result in a general form. Besides, the problem with the majority of data on the Web that is in this form now is that it is difficult to use on a large scale, because there is no global system for publishing data in such a way as it can be easily processed by anyone. The vision of Semantic Web envisage the web enriched with several domain ontologies, which specify formal semantic of data and that can be used for different intelligent service, like information research, retrieval, and transformation. The suitable methodology to be used is waterfall model as it provides flexibility on developing this semantic web.

ACKNOWLEDGEMENT

First and foremost, the author would like to thank the Almighty Allah for His blessings and wishes to take this opportunity to acknowledge and recognize the contributions of the individuals who have helped the author throughout this project.

The author extends his profound gratitude to Mrs. Mazeyanti Mohd Ariffin (project supervisor) who have showered the author with a lot of information and knowledge pertaining to this project. Without their relentless help, the author would not have been able to complete this course successfully. It has truly been an amazing experience to be under their guidance and supervision.

The author also recognizes the effort of all the staff and lecturers from the Information System Department in assisting the author throughout the success of this project.

Last but not least, the author would like to record his appreciation to the other individuals, who will identify themselves in this acknowledgement, for their contribution towards the realization of this project.

TABLE OF CONTENT

CERTIFICATION OF APPROVAL	I
CERTIFICATION OF ORIGINALITY	II
ABSTRACT	III
ACKNOWLEDGEMENT	IV
LIST OF FIGURES	VI
ABBREVIATION AND NOMENCLATURES	VII
CHAPTER 1:	
INTRODUCTION.....	1
1.1 Background Study.....	1
1.2 Problem Statement.....	2
1.3 Objective and Scope of Study.....	3
1.4 Significant of Project.....	3
1.5 Feasibility of Project Within Time Frame.....	5
CHAPTER 2:	
LITERATURE REVIEW AND THEORY.....	6
2.1 Relationship to the world wide web.....	6
2.2 Component of Semantic Web.....	8
2.2.1 Basic Assertion Model.....	10
2.2.2 Basic RDF Model.....	13
2.3 Semantic Web Ontology.....	18
2.4 Knowledge Representation.....	20
2.4.2 Agent.....	22
CHAPTER 3:	
METHODOLOGY/PROJECTWORK.....	25
3.1 Project Approach.....	25
3.1.1 Requirement And System Analysis.....	27
3.1.2 System Analysis And Specification Phase.....	27
3.1.3 Design And Implementation.....	28
3.1.4 Integration.....	28

3.1.5 Maintenance.....	29
CHAPTER 4:	
RESULT AND DISCUSSION.....	30
4.1 Result And Discussion.....	30
4.2 Semantic Web Framework.....	30
4.3 Contribution.....	32
4.4 Semantic Security.....	32
4.4.1 Control Over Changing.....	32
4.4.2 Control Over Storing.....	33
4.5 Query Protocol.....	33
4.6 Interface.....	33
4.6.1 Semantic Web Browser.....	33
CHAPETR 5:	
CONCLUSION AND RECOMMENDATION.....	35
REFERENCE.....	36

LIST OF FIGURES

Figure 1 : Key Map loop

Figure2: Simple note and arc diagram

Figure 3: Property with structure value

Figure 4: Structured value with identifier

Figure 5: Project Methodology

Figure 6: Semantic web framework

ABBREVIATION AND NOMENCLATURES

URI	: Uniform Resource Identifier
HTML	: Hyper Text Markup Language
RDF	: Resource Description Framework
XML	: Extended Markup Language
OWL	: Web Ontology Language
HTTP	: Hyper Text Transfer Protocol
OMS	: Ontology Management System

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND OF STUDY

Semantic web is mesh information linked up in such way as to be easily processable by machine on a global scale. We will think of it as an efficient way of representing data on a world wide web or as globally linked database. Beside Semantic Web can also be as a huge engineering solution. This is because, we will find that it is easy to publish data in repurposable form when using semantic web.

The semantic web will be integrated with a knowledge management agent, in which it will be as the platform for the knowledge agent to classify all the information before it been stored in the database.

Semantic Web service will always deal with metadata. This is because metadata-based profile can be used to change a service the website can provide and allow the developers to program presentation and tailoring of data. Therefore, this concept is necessary to be applied in this project, as the main purpose of the project is to classify the output from the search result via the search engine. Beside the Semantic Web are generally build on syntax which use URI's (Uniform Resource Identifier) 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 World Wide Web using a set of particular syntaxes developed especially for the task. With the help of the

current agent that we have, the hits will go through multiple search engines and this will produce a various kind of output.

1.2 PROBLEM STATEMENT

Generally:

- Data is generally hidden away in HTML files is often useful in some context, but not in others
- The current problem with majority data on the web that is in HTML form is that it is difficult to see them on a larger scale.
- There are no such system that can be used to publish the data that can be used in a large scale and that can be process by the machine

Extended to the general problem statement that had been mention above, we had encounter the other problem that been faced by the user in manipulating and use of the information. When they are browsing through the internet and looking for some information via the search engine. This will drag user to analyze all the output been produce by the search engine in order to satisfied their need of information. This is because the search engine did not have the knowledge management agent to perform the specific task, especially in classifying the output. They did hand in good information to the user, but they fail to integrate the knowledge management with the data that they have. This will prohibit user to easily process the information that they already have.

1.3 OBJECTIVES AND SCOPE OF STUDY

The objectives of this project are:

- To study and proof that the system could provide a “well define information” to the user
- To build a search engine that will process the hits according to the user searching preferences.
- To determine that the system will provide the easiness to the user in performing searching for the information in the internet
- To determine that the system will shorten the searching time since the system will crawl directly to the source of information.

1.4

SIGNIFICANT OF PROJECT

As been mentioned above, the project is really meant if it successfully integrates the knowledge management agent with the Semantic Web service. If this happen, then we can have a search engine that can fully satisfied the user needs and supply the information in a very good manner. Besides, while search engines which index HTML pages find many answers to searches and cover a huge part of the Web, then return many inappropriate answers. There is no notion of "correctness" to such searches. By contrast, logical engines have typically been able to restrict their output

to that which is provably correct answer, but have suffered from the inability to rummage through the mass of intertwined data to construct valid answers. The combinatorial explosion of possibilities to be traced has been quite intractable.

However, the scale upon which search engines have been successful may force us to reexamine our assumptions here. If an engine of the future combines a reasoning engine with a search engine, it may be able to get the best of both worlds, and actually be able to construct proofs in a certain number of cases of very real impact. It will be able to reach out to indexes which contain very complete lists of all occurrences of a given term, and then use logic to weed out all but those which can be of use in solving the given problem. So while nothing will make the combinatorial explosion go away, many real life problems can be solved using just a few (say two) steps of inference out on the wild web, the rest of the reasoning being in a realm in which proofs are give, or there are constrains and well understood computable algorithms. I also expect a string commercial incentive to develop engines and algorithms which will efficiently tackle specific types of problem. This may involve making caches of intermediate results much analogous to the search engines' indexes of today. Though there will still not be a machine which can guarantee to answer arbitrary questions,

the power to answer real questions which are the stuff of our daily lives and especially of commerce may be quite remarkable.

1.5 FEASIBILITY OF PROJECT WITHIN TIME FRAME

The project consists of 4 main parts:

- Gathering information which is defining a Semantic Web.
- Gathering information related to the tools and technique that been used in the development of semantic web.
- Conducting a research on the framework of the development of Semantic Web
- The implementation of Semantic Web

Based on the available resources to perform the above tasks, it is feasible to complete the project within the given time frame.

CHAPTER 2

LITERATURE REVIEW AND THEORY

2.1 RELATIONSHIP TO THE WORLD WIDE WEB

Currently, the World Wide Web is based primarily on documents written in HTML, a language that is useful for describing, with an emphasis on visual presentation, a body of structured text interspersed with multimedia objects such as images and interactive forms. HTML has limited ability to classify the blocks of text on a page, apart from the roles they play in a typical document's organization and in the desired visual layout.

For example, with HTML and a tool to render it (perhaps Web browser software, perhaps another user agent), one can create and present a page that lists items for sale. The HTML of this catalog page can make simple, document-level assertions such as "this document's title is 'Widget Superstore'". But there is no capability within the HTML itself to unambiguously assert that, say, item number X586172 is an Acme Gizmo with a retail price of €199, or that it is a consumer product. Rather, HTML can only say that the span of text "X586172" is something that should be positioned near "Acme Gizmo" and "€199", etc. There is no way to say "this is a catalog" or even to establish that "Acme Gizmo" is a kind of title or that "€199" is a price. There is also no way to express that these pieces of information are bound together in describing a discrete item, distinct from other items perhaps listed on the page.

[(Example is taken from wikipedia.org)]

The Semantic Web addresses this shortcoming, using the descriptive technologies RDF and OWL, and the data-centric, customizable markup language XML. These technologies are combined in order to provide descriptions that supplement or replace the content of Web documents. Thus, content may manifest as descriptive data stored in Web-accessible databases, or as markup within documents (particularly, in XHTML interspersed with XML, or, more often, purely in XML, with layout/rendering cues stored separately). The machine-readable descriptions allow content managers to add meaning to the content, thereby facilitating automated information gathering and research by computers.

2.2 COMPONENT OF SEMANTIC WEB

From the reaseach done by the writer, I shows that Semantic Web is comprised of the standards and tools of XML, XML Schema, RDF, RDF Schema and OWL. The OWL Web Ontology Language Overview describes the function and relationship of each of these components of the Semantic Web:

- XML provides a surface syntax for structured documents, but imposes no semantic constraints on the meaning of these documents.
- XML Schema is a language for restricting the structure of XML documents.
- RDF is a simple data model for referring to objects ("resources") and how they are related. An RDF-based model can be represented in XML syntax.
- RDF Schema is a vocabulary for describing properties and classes of RDF resources, with a semantics for generalization-hierarchies of such properties and classes.
- OWL adds more vocabulary for describing properties and classes: among others, relations between classes (e.g. disjointness), cardinality (e.g. "exactly one"), equality, richer typing of properties, characteristics of properties (e.g. symmetry), and enumerated classes.

The intent is to enhance the usability and usefulness of the Web and its interconnected resources through:

- documents "marked up" with semantic information (an extension of the HTML <meta> tags used in today's Web pages to supply information for Web search

engines using web crawlers). This could be machine-readable information about the human-readable content of the document (such as the creator, title, description, etc., of the document) or it could be purely metadata representing a set of facts (such as resources and services elsewhere in the site). (Note that *anything* that can be identified with a *Uniform Resource Identifier* (URI) can be described, so the semantic web can reason about people, places, ideas, cats etc.)

- common metadata vocabularies (ontologies) and maps between vocabularies that allow document creators to know how to mark up their documents so that agents can use the information in the supplied metadata (so that Author in the sense of 'the Author of the page' won't be confused with Author in the sense of a book that is the subject of a book review).
- automated agents to perform tasks for users of the Semantic Web using this metadata
- web-based services (often with agents of their own) to supply information specifically to agents (for example, a Trust service that an agent could ask if some online store has a history of poor service or spamming).

The primary facilitators of this technology are URIs (which identify resources) along with XML and namespaces. These, together with a bit of logic, form RDF, which can be used to say anything about anything. As well as RDF, many other technologies such as Topic Maps and pre-web artificial intelligence technologies are likely to contribute to the Semantic Web.

A popular application of the Semantic Web is Friend of a Friend (or FoaF), which describes relationships among people and other agents in terms of RDF.

2.2.1 BASIC ASSERTION MODEL

When looking at a possible formulation of a universal Web of semantic assertions, the principle of minimalist design requires that it be based on a common model of great generality. If the common model is general, any prospective application can be mapped onto the model. This is what the Resource Description Framework deal with. The basic model contains just the concept of an assertion, and the concept of quotation (making assertions about assertions). This is introduced because it will be needed later anyway and most of the initial RDF applications are for data about data ("metadata") in which assertions about assertions are basic, even before logic. This is due to the target applications of RDF, assertions are part of a description of some resource, that resource is often an implicit parameter and the assertion is known as a property of a resource. As far as mathematics goes, the language at this point has no negation or implication, and is therefore very limited. Given a set of facts, it is easy to say whether a proof exists or not for any given question, because neither the facts nor the questions can have enough power to make the problem intractable. Applications at this level are very numerous. Most of the applications for the representation of metadata can be handled by RDF at this level. The representation of data is typically simple: not languages for expressing queries or inference rules. RDF documents at this level do not have great power, and sometimes it is less than evident why one should bother to map an application in RDF. The answer is that we expect this data, while limited and simple within an application, to be combined, later, with data from other applications into a Web. Applications which run over the whole web must be able to use a common framework for combining information from all these applications. For example, access control logic may use a combination of privacy and group membership and data type information to actually allow or deny access. Queries may later allow powerful logical expressions referring to data from domains

in which, individually, the data representation language is not very expressive. The purpose of this document is partly to show the plan by which this might happen.

The metro map below shows a key loop in the semantic web. The Web part, on the left, shows how a URI is, using HTTP, turned into a representation of a document as a string of bits with some MIME type. It is then parsed into XML and then into RDF, to produce an RDF graph or, at the logic level, a logical formula. On the right hand side, the Semantic part, shows how the RDF graph contains a reference to the URI. It is the trust from the key, combined with the meaning of the statements contained in the document, which may cause a Semantic Web engine to dereference another URI.

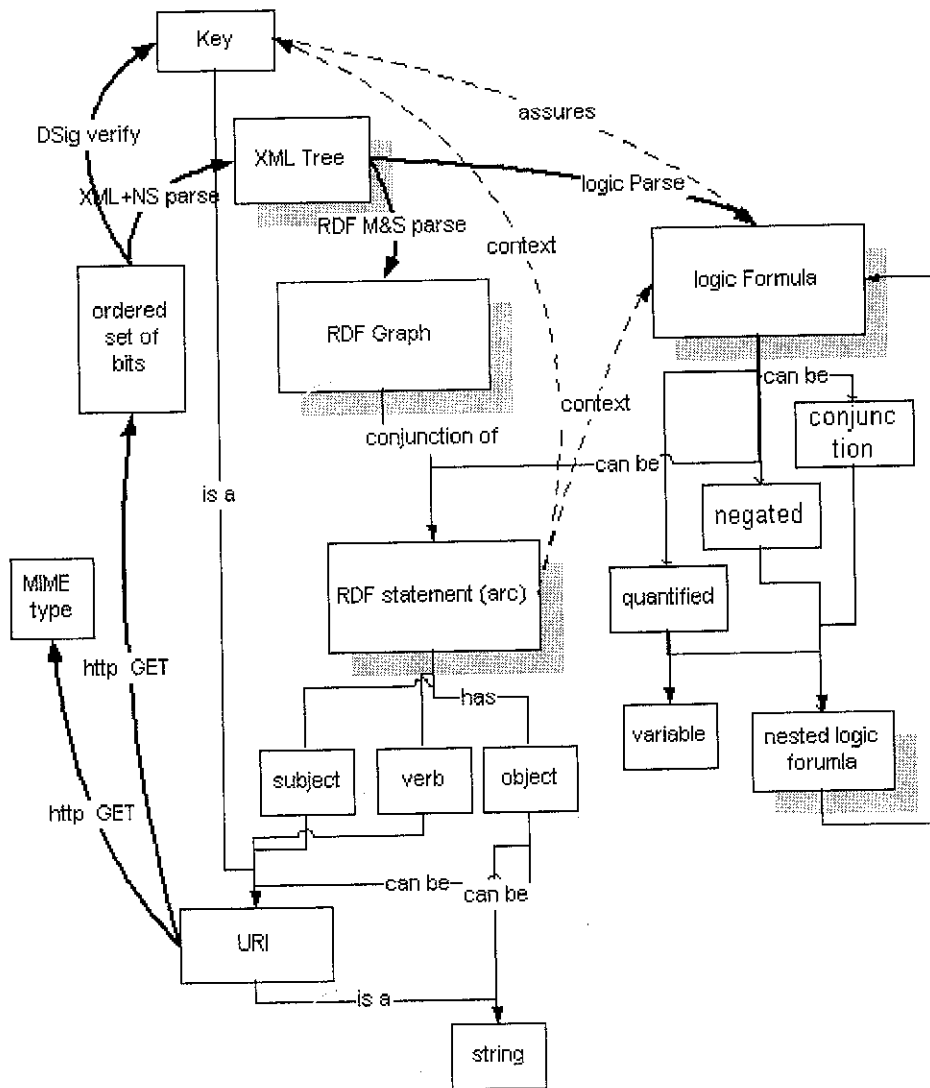


Figure 1: Key Map loop

2.2.2 BASIC RDF MODEL

The foundation of RDF is a model for representing named properties and property values. The RDF model draws on well-established principles from various data representation communities. RDF properties may be thought of as attributes of resources and in this sense correspond to traditional attribute-value pairs. RDF properties also represent relationships between resources and an RDF model can therefore resemble an entity-relationship diagram. (More precisely, RDF Schemas — which are themselves instances of RDF data models — are ER diagrams.) In object-oriented design terminology, resources correspond to objects and properties correspond to instance variables.

The RDF data model is a syntax-neutral way of representing RDF expressions. The data model representation is used to evaluate equivalence in meaning. Two RDF expressions are equivalent if and only if their data model representations are the same. This definition of equivalence permits some syntactic variation in expression without altering the meaning.

The basic data model consists of three object types:

- Resources** All things being described by RDF expressions are called *resources*. A resource may be an entire web page; such as the HTML document "http://www.elearning.edu.my/Overview.html" for example. A resource may be a part of a web page; e.g. a specific HTML or XML element within the document source. A resource may also be a whole collection of pages; e.g. an entire web site. A resource may also be an object that is not directly accessible via the Web; e.g. a printed book. Resources are always named by URIs plus optional anchor IDs. Anything can have a URI; the extensibility of URIs allows the introduction of identifiers for any entity imaginable.
- Properties** A *property* is a specific aspect, characteristic, attribute, or relation used to describe a resource. Each property has a specific meaning, defines its permitted values, the types of resources it can describe, and its relationship with other properties. This document does not address how the characteristics of properties are expressed; for such information.
- Statements** A specific resource together with a named property plus the value of that property for that resource is an RDF *statement*. These three individual parts of a statement are called, respectively, the *subject*, the *predicate*, and the *object*. The object of a statement (i.e. the property value) can be another resource or it can be a literal; i.e. a resource (specified by a URI) or a simple string or other primitive datatype defined

by XML. In RDF terms, a *literal* may have content that is XML markup but is not further evaluated by the RDF processor. There are some syntactic restrictions on how markup in literals may be expressed; see

Examples

Resources are identified by a *resource identifier*. A resource identifier is a URI plus an optional anchor id. For the purposes of this section, properties will be referred to by a simple name.

Consider as a simple example the sentence:

Shahrul Anuar is the creator of the resource
<http://www.elearning.edu.my/Home/Anuar>

This sentence has the following parts:

Subject (Resource)	http://www.elearning.edu.my/Home/Anuar
Predicate (Property)	Creator
Object (literal)	"Shahrul Anuar"

In this document we will diagram an RDF statement pictorially using directed labeled graphs (also called "nodes and arcs diagrams"). In these diagrams, the nodes (drawn as ovals) represent resources and arcs represent named properties. Nodes that represent string literals will be drawn as rectangles. The sentence above would thus be diagrammed as:

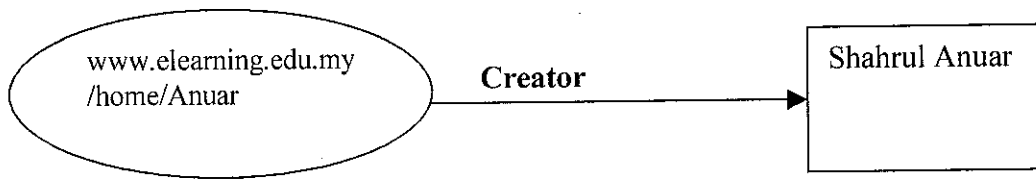


Figure 2: Simple note and arc diagram

Now, consider the case that we want to say something more about the characteristics of the creator of this resource. In prose, such a sentence would be:

The individual whose name is Shahrul Anuar, email <nofx2001@yahoo.com>, is the creator of <http://www.elearning.edu.my/Home/Anuar>

The intention of this sentence is to make the value of the Creator property a structured entity. In RDF such an entity is represented as another resource. The sentence above does not give a name to that resource; it is anonymous, so in the diagram below we represent it with an empty oval:

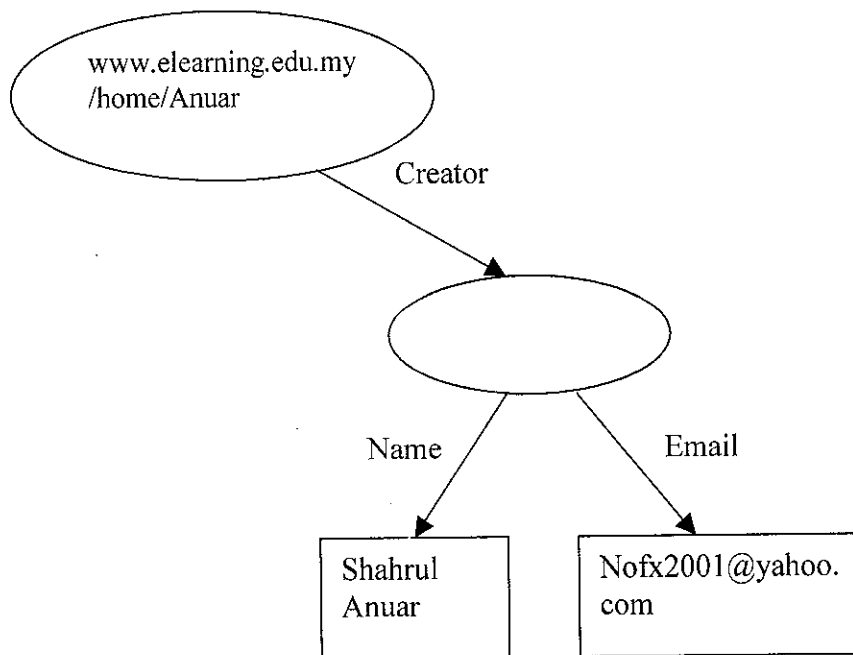


Figure 3: Property with structure value

The structured entity of the previous example can also be assigned a unique identifier. The choice of identifier is made by the application database designer. To continue the example, imagine that an employee id is used as the unique identifier for a "person" resource. The URIs that serves as the unique keys for each employee (as defined by the organization) might then be something like <http://www.elearning.edu.my/staffId/85740>. Now we can write the two sentences:

The individual referred to by employee id 85740 is named Shahrul Anuar and has the email address nofx2001@yahoo.com. The resource <http://www.elearning.edu.my/Home/Anuar> was created by this individual.

The RDF model for these sentences is:

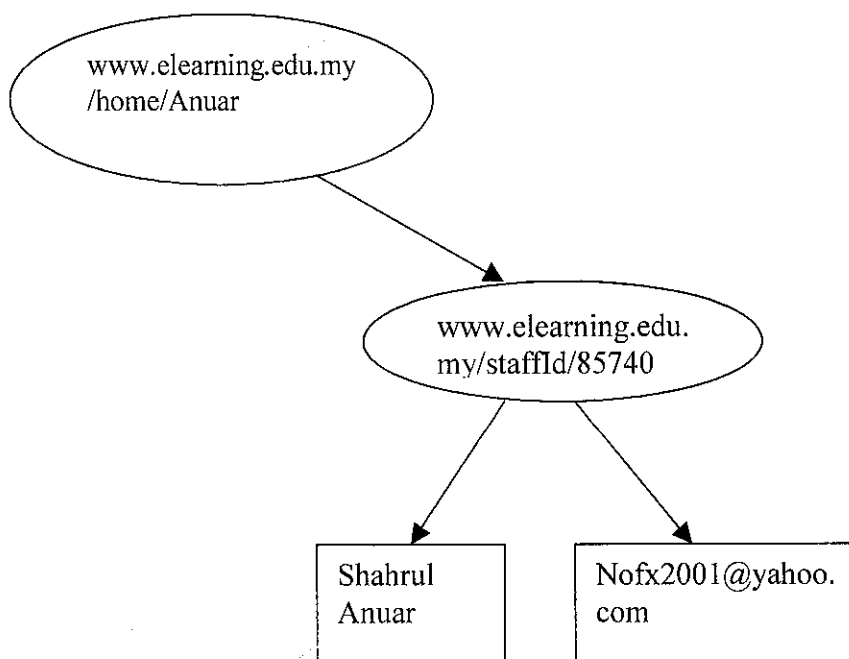


Figure 4: Structured value with identifier

Note that this diagram is identical to the previous one with the addition of the URI for the previously anonymous resource. From the point of view of a second application querying this model, there is no distinction between the statements made in a single sentence and the statements made in separate sentences.

2.3 SEMANTIC WEB ONTOLOGIES

In philosophy, an ontology is a theory about the nature of existence, of what types of things exist; ontology as a discipline studies such theories. Artificial-intelligence and Web researchers have co-opted the term for their own jargon, and for them an ontology is a document or file that formally defines the relations among terms. The most typical kind of ontology for the Web has a taxonomy and a set of inference rules.

The taxonomy defines classes of objects and relations among them. For example, an *address* may be defined as a type of *location*, and *city codes* may be defined to apply only to *locations*, and so on. Classes, subclasses and relations among entities are a very powerful tool for Web use. We can express a large number of relations among entities by assigning properties to classes and allowing subclasses to inherit such properties. If *city codes* must be of type *city* and cities generally have Web sites, we can discuss the Web site associated with a *city code* even if no database links a city code directly to a Web site.

Inference rules in ontologies supply further power. An ontology may express the rule if a city code is associated with a state code, and an address uses that city code, then that address has the associated state code. A program could then readily deduce, for

instance, that a University Technology Of Petronas address, being in Tronoh, must be in Perak

State, which is in the Malaysia, and therefore should be formatted to Malaysia standards. The computer does not truly understand any of this information, but it can now manipulate the terms much more effectively in ways that are useful and meaningful to the human user.

With ontology pages on the Web, solutions to terminology (and other) problems begin to emerge. The meaning of terms or XML codes used on a Web page can be defined by pointers from the page to an ontology. Of course, the same problems as before now arise if writer point to an ontology that defines *addresses* as containing a *zip code* and you point to one that uses *postal code*. This kind of confusion can be resolved if ontologies (or other Web services) provide equivalence relations: one or both of our ontologies may contain the information that my zip code is equivalent to your postal code.

The program, using distinct URIs for different concepts of address, will not confuse them and in fact will need to discover that the concepts are related at all. The program could then use a service that takes a list of *postal addresses* (defined in the first ontology) and converts it into a list of physical *addresses* (the second ontology) by recognizing and removing post office boxes and other unsuitable addresses. The structure and semantics provided by ontologies make it easier for an entrepreneur to provide such a service and can make its use completely transparent.

Ontologies can enhance the functioning of the Web in many ways. They can be used in a simple fashion to improve the accuracy of Web searches—the search program can look for only those pages that refer to a precise concept instead of all the ones using ambiguous keywords.

In addition, this markup makes it much easier to develop programs that can tackle complicated questions whose answers do not reside on a single Web page. Suppose you wish to find the Ms. Cook you met at a trade conference last year. You do not remember her first name, but you remember that she worked for one of your clients and that her son was a student at your alma mater. An intelligent search program can sift through all the pages of people whose name is "Cook" (sidestepping all the pages relating to cooks, cooking, the Cook Islands and so forth), find the ones that mention working for a company that's on your list of clients and follow links to Web pages of their children to track down if any are in school at the right place.

2.4 KNOWLEDGE REPRESENTATION

For the semantic web to function, computers must have access to structured collections of information and sets of inference rules that they can use to conduct automated reasoning. Artificial-intelligence researchers have studied such systems since long before the Web was developed. Knowledge representation, as this technology is often called, is currently in a state comparable to that of hypertext before the advent of the Web: it is clearly a good idea, and some very nice demonstrations exist, but it has not yet changed the world. It contains the seeds of important applications, but to realize its full potential it must be linked into a single global system.

Traditional knowledge-representation systems typically have been centralized, requiring everyone to share exactly the same definition of common concepts such as "parent" or "vehicle." However, central control is stifling, and increasing the size and scope of such a system rapidly becomes unmanageable. [(Miguel Salmeron 2001)]

Moreover, these systems usually carefully limit the questions that can be asked so that the computer can answer reliably or answer at all. The problem is reminiscent of Gödel's theorem from mathematics: any system that is complex enough to be useful also encompasses unanswerable questions, much like sophisticated versions of the basic paradox "This sentence is false." To avoid such problems, traditional knowledge-representation systems generally each had their own narrow and idiosyncratic set of rules for making inferences about their data. Semantic Web researchers, in contrast, accept that paradoxes and unanswerable questions are a price that must be paid to achieve versatility. We make the language for the rules as expressive as needed to allow the Web to reason as widely as desired. This philosophy is similar to that of the conventional Web: early in the Web's development, detractors pointed out that it could never be a well-organized library; without a central database and tree structure, one would never be sure of finding everything. They were right. However, the expressive power of the system made vast amounts of information available, and search engines (which would have seemed quite impractical a decade ago) now produce remarkably complete indices of a lot of the material out there. The challenge of the Semantic Web, therefore, is to provide a language that expresses both data and rules for reasoning about the data and that allows rules from any existing knowledge-representation system to be exported onto the Web.

Adding logic to the Web—the means to use rules to make inferences, choose courses of action and answer questions—is the task before the Semantic Web community at the moment. A mixture of mathematical and engineering decisions complicates this task. The logic must be powerful enough to describe complex properties of objects but not so powerful that agents can be tricked by being asked to consider a paradox. Fortunately, a large majority of the information we want to express is along the lines of "a hex-head bolt is a type of machine bolt," which is readily written in existing languages with a little extra vocabulary.

2.4.1 AGENT

The real power of the Semantic Web will be realized when people create many programs that collect Web content from diverse sources, process the information and exchange the results with other programs. The effectiveness of such software agents will increase exponentially as more machine-readable Web content and automated services (including other agents) become available. The Semantic Web promotes this synergy: even agents that were not expressly designed to work together can transfer data among themselves when the data come with semantics.

An important facet of agents' functioning will be the exchange of "proofs" written in the Semantic Web's unifying language (the language that expresses logical inferences made using rules and information such as those specified by ontologies). For example, suppose 1919 contact information has been located by an online service, and to your great surprise, it places in Johannesburg. Naturally, you want to check this, so your computer asks the service for a proof of its answer, which it promptly provides by translating its internal reasoning into the Semantic Web's unifying language. An inference engine in your computer readily verifies that this 1919 Restaurant indeed matches the one you were seeking, and it can show you the relevant Web pages if you still have doubts. Although they are still far from plumbing

the depths of the Semantic Web's potential, some programs can already exchange proofs in this way, using the current preliminary versions of the unifying language.

Another vital feature will be digital signatures, which are encrypted blocks of data that computers and agents can use to verify that the attached information has been provided by a specific trusted source. You want to be quite sure that a statement sent to your accounting program that you owe money to an online retailer is not a forgery generated by the computer-savvy teenager next door. Agents should be skeptical of assertions that they read on the Semantic Web until they have checked the sources of information. Many automated Web-based services already exist without semantics, but other programs such as agents have no way to locate one that will perform a specific function. This process, called service discovery, can happen only when there is a common language to describe a service in a way that lets other agents "understand" both the function offered and how to take advantage of it. Services and agents can advertise their function by, for example, depositing such descriptions in directories analogous to the Yellow Pages.

Some low-level service-discovery schemes are currently available, such as Microsoft's Universal Plug and Play, which focuses on connecting different types of devices, and Sun Microsystems's Jini, which aims to connect services. These initiatives, however, attack the problem at a structural or syntactic level and rely heavily on standardization of a predetermined set of functionality descriptions. Standardization can only go so far, because we cannot anticipate all possible future needs.

The Semantic Web, in contrast, is more flexible. The consumer and producer agents can reach a shared understanding by exchanging ontologies, which provide the vocabulary needed for discussion. Agents can even "bootstrap" new reasoning capabilities when they discover new ontologies. Semantics also makes it easier to take advantage of a service that only partially matches a request.

A typical process will involve the creation of a "value chain" in which subassemblies of information are passed from one agent to another, each one "adding value," to construct the final product requested by the end user. Make no mistake: to create complicated value chains automatically on demand, some agents will exploit artificial-intelligence technologies in addition to the Semantic Web. However, the Semantic Web will provide the foundations and the framework to make such technologies more feasible.

In the next step, the Semantic Web will break out of the virtual realm and extend into our physical world. URIs can point to anything, including physical entities, which means we can use the RDF language to describe devices such as cell phones and TVs. Such devices can advertise their functionality, what they can do and how they are controlled, much like software agents. Being much more flexible than low-level schemes such as Universal Plug and Play, such a semantic approach opens up a world of exciting possibilities.

CHAPTER 3

METHODOLOGY/PROJECT WORK

3.1 PROJECT APPROACH

Project approach towards developing Semantic Web integrated with Knowledge Management agent will be based on the project development schedule prepared by the final year project committee. Expected deliverables within given time frame is to be met to ensure project continuity and accomplishment.

The phase included in the method will list down the steps that the author will take as a guide throughout the project development. The methodology of the project is shown in figure 5.

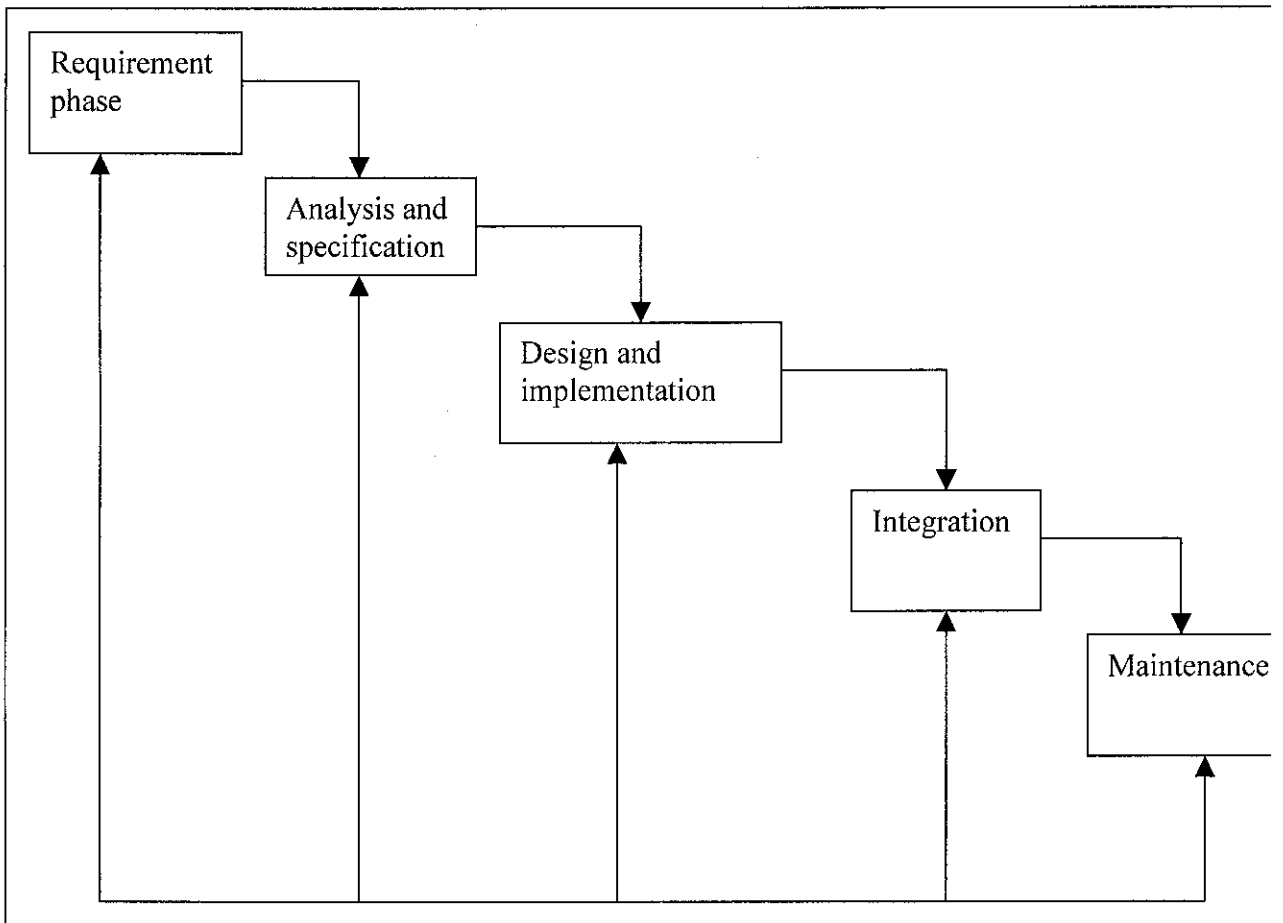


Figure 5: Project Methodology

3.1.1 REQUIREMENT ANALYSIS

For this project there are several requirements needed in order to complete it. As the project will integrate the Knowledge Management agent with a Semantic Web means that there are two different systems and each of them has its own requirements. As the semantic web will act as a search engine, therefore first the writer needs to analyze how does the standard or normal search engine work. It is a different thing between a standard search engine and semantic web. By knowing how does the standard search engine work and process all the hits, we could enhance it and apply the concept in Semantic Web environment. For the requirements analysis, the writer needs to know about the requirements needed to develop a Semantic Web. Then, gather other information on the agent that is going to be used.

3.1.2 SYSTEM ANALYSIS AND SPECIFICATION PHASE

During this phase, it will take all effort to analyze everything about both the Semantic Web and also the agent. This analysis will cover everything including the tools needed for development, the language that is going to be used, the system's framework, the resources and so on. Besides, this phase also will analyze about the system flow, which describes how the user will interact with the system. The specification is also an important criterion that should be checked and determined before we start developing this system. This is to enable us to assign any important variable, which may lead to the failure of the system if we miss the steps. The analyzing and doing the specification is also applied to the agent that the system is going to use. As mentioned above, we need to know how the system is going to represent the knowledge that has been sent to them.

3.1.3 DESIGN AND IMPLEMENTATION

Designing phase is where all the requirement and information which is available for the project been combine. This is where the algorithm been include in the system. Besides in this phase, it need to be ensure that the development are according to the framework that had already been define. The system will be design phase by phase and it will start with the designing of semantic web including built the related object and classes, defining the relations among them and followed by the knowledge management agent. All the information such the ontology or taxonomy will be implement according to the need of the system. For the agent, the design phase will focus on how they are going to exchange the proof and how they are going to communicate in the system.

3.1.4 INTERGRATION

Once completing the designing and implementation phase, then the system will be integrated (the agent and the Semantic Web). The system will be test in order to confirm that it is compatible and if there is an error occurs, the system will be segregate to fix the problem and then it will be re-integrate.

3.1.5 MAINTAINENCE

Maintenances need to be done when system is already been used. This is to ensure that the system will always work properly. Besides, any upgrading of the system will also be done during this phase.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 RESULTS AND DISCUSSION

This chapter reports the details of the system and also the finding during the development of the system.

4.2 SEMANTIC WEB FRAMEWORK

The framework for the Semantic Network Technologies software is implemented using a commercially available ontology management system (OMS) along with a collection of developed tools that provide synergistic services as shown below:

Semantic Framework

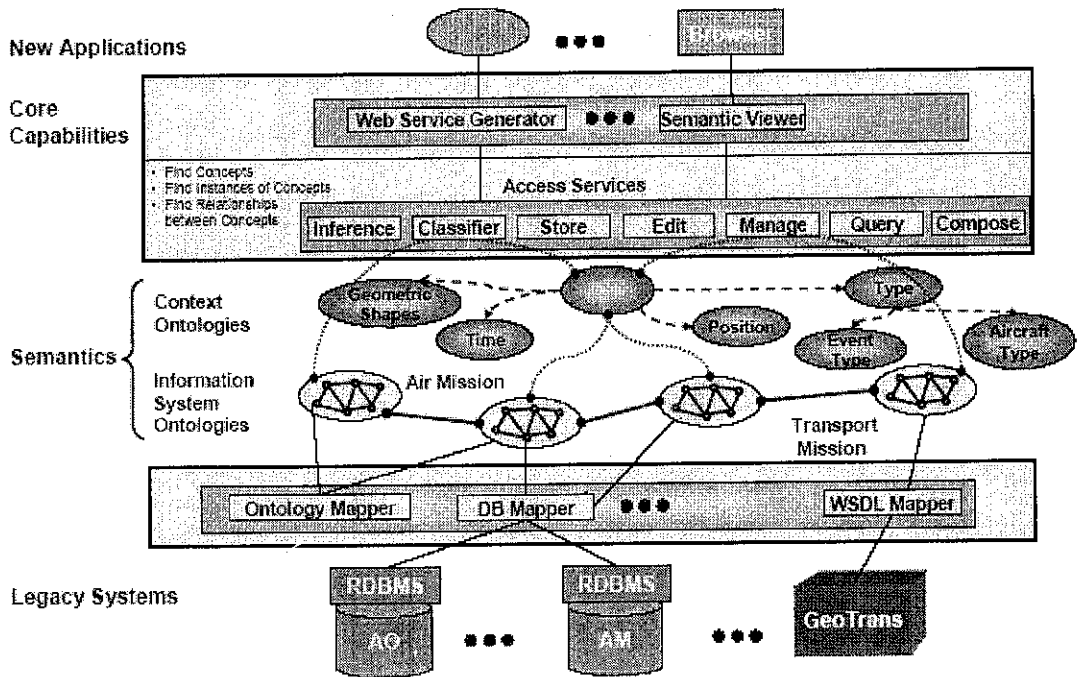


Figure 6: Semantic web framework

[(taken from Geotrans.com)]

4.2 THE CONTRIBUTION

The knowledge on the Semantic Web is an aggregate of contributions from many sources, much as the Web is an aggregate of many web sites. The answers to a question you ask, like the results of a web search, will depend to some degree on who you trust, which systems are working at the time you ask, and what kind of search techniques are being used.

As you browse information, a good user interface will make it easy to correct inaccuracies and add your own knowledge as you like. Your additions will be stored in various configurable ways and can be kept private or published as openly as you choose.

4.3 SEMANTIC SECURITY

This is the important feature that will control the access to the in going and the out going of the data in Semantic Web. Its comprises of 2 access controls which are:

4.3.1 CONTROL OVER CHANGING

To change some data on the Semantic Web, you don't need the permission of the author of the data you're changing; you need the trust of the reader.

4.3.2 CONTROL OVER STORING

Sometimes user want to change what is store, either just because they need someplace to store it, or because they want it to be published from a particular address.

4.4 QUERY PROTOCOLS

If user require query-answering-agents to implement a sufficient logic then a query protocol is trivial; it just requires an class of objects which, when described, cause specific (results) information to be sent along an implicit return path.

4.5 INTERFACES

4.5.1 SEMANTIC WEB BROWSER

An HTTP URI with no fragment should generally identify a contribution to the web, a collection of knowledge. It probably has an HTML rendering, which is that knowledge put in a form that's easy for humans to understand. It should also have a structural (RDF) form, which is easy for humans and machines to understand. On good sites, with good browsers, the HTML will become more and more useless.

A URI with a fragment identifier is taken to denote an object described in the knowledge base identified by the base URI. This is a slight stretch from its conventional usage, but fragment identifiers are in fact usually used to name something discussed on the page. (Sometimes they just name a section, but in that case usually the page has a table of contents and a corresponding knowledge base might similarly identify contained collections of knowledge. But perhaps that's overreaching.)

A Semantic Web browser needs (1) an identifier of the thing you want information about, or the collection of things, or the collection of information, and (2) a configuration of what sources to use and what algorithms to use to find more sources, and (3) info about how to store changes you make, (4) appearance preferences.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

As for the conclusion, it is possible to have Semantic Web that is integrated with a knowledge management agent. In addition, with the existing of this system, it will help user to access a very high level of data. Besides, user also will obtain a specific result base on what they are searching for.

The component that been included in Semantic Web such, the ontology that will work on the data exchange and also the RDF which related to the resource and metadata will ensure that this Semantic Web will generate and gather as much as information that is available for the user, so it could widen the searching scope and produce as much result as they can. The integrated agent will work on the management of the data, help user to get a specific data and will excluded all the junk data, which is not necessary to appear on the result page. Having such search engine will give many advantage and benefit to the user. Instead of having a shorter time to access to the information available, they are also been fed with the information that comes from many sites and resources.

For the recommendation, the system can be improve if the system could improve the way it identify the resource or if it can specifically grab the data and metadata base on the hits from the user. Upgrading the agent will also can add value to the system. Maybe we could have an agent that could specify all the result according to the classes.

I hope that this research can be continued in the future as Semantic Web with integrated agent can be considered as a new technology in Malaysia and still did not widely developed.

REFERENCES

- Michael C. Daconta, Leo J. Obrst, Kevin T. Smith: *The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management*, John Wiley & Sons, ISBN 0-471-43257-1
- Dieter Fensel, Wolfgang Wahlster, Henry Lieberman, James Hendler: *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*, MIT Press, ISBN 0-262-06232-1
- Steffen Staab, Rudi Studer: *Handbook on Ontologies*, Springer Verlag, Heidelberg, 2004, ISBN 3-540-40834-7
- Vladimir Geroimenko, Chaomei Chen: *Visualizing the Semantic Web*, Springer Verlag, ISBN 1-85233-576-9
- John Davies, Dieter Fensel, Frank van Harmelen: *Towards the Semantic Web: Ontology-Driven Knowledge Management*, John Wiley & Sons, ISBN 0-470-84867-7
- Grigoris Antoniou, Frank van Harmelen: *A Semantic Web Primer*, The MIT Press, ISBN 0-262-01210-3
- Jeffrey T. Pollock, Ralph Hodgson: *Adaptive Information: Improving Business through Semantic Interoperability, Grid Computing, and Enterprise Integration*, John Wiley, ISBN 0-471-48854-2
- <http://www.w3c.org>
- <http://www.agentland.com>
- James Hendler, 2001 "Paper on Agents and the Semantic Web", Department of Computer Science University of Maryland