Using Semantic Web to Provide Intelligent Advice on University Admissions

Francis King Hei KWONG
City University of Hong Kong
Department of Computer Science
Tat Chee Avenue, Kowloon
Hong Kong SAR
50190500@student.cityu.edu.hk

Hon Wai CHUN
City University of Hong Kong
Department of Computer Science
Tat Chee Avenue, Kowloon
Hong Kong SAR
andy.chun@cityu.edu.hk

Abstract

This paper describes a prototype application that investigates benefits of using Semantic Web to improve information access and provide intelligent advice to students in selecting the most appropriate universities and subjects to apply to. Without Semantic Web technology, students need to manually browse through pages and pages of information in Web sites of different Universities to find a list of universities, departments and programs that they qualify to apply for. Our prototype demonstrates that Semantic Web can automate and streamline this whole process. All the students need to do is enter their qualifications and a list will be generated automatically. Of course, this is assuming that most universities have the admissions requirements in RDF format. Fortunately, since there are only eight universities in Hong Kong, the use of a shared ontology based on RDF and DAML+OIL to describe admissions requirements is not totally impossible. One of the objectives of our research is to design such an ontology and to test its limits by encoding admission requirements for several of our local universities into our prototype. This paper also presents a new "explore by class" search algorithm we developed to optimize search performances in our university Semantic Web problem domain.

1. Introduction

Semantic Web is the next generation of World Wide Web. The significance of the Semantic Web is that information on the web will be machine-understandable [1], so that machine reasoning on web page information is possible. The Semantic Web is a web of data, in some ways like a global database [2]. Semantic Web organizes information in a well defined structure and creates an environment where software agent can readily carry out sophisticated tasks on the web for users. In this paper, we explore how such a software agent might be developed to retrieve precise university admission requirements from different university websites, match them with high school student's qualifications and present a filtered list of candidate universities for a student to consider applying to. The main objective of this project is to assess the feasibility of using Semantic Web as well as analyze its usefulness for such an application. To achieve these objectives, we designed and developed a Semantic Web-based "My First Uni" (MFU) application.

2. The Problem

High school graduates in Hong Kong who are continuing their studies in a university will have to submit a list of potential bachelor degree programs that they are interested in to the Joint University Programmes Admissions System (JUPAS). JUPAS is a semi-government organization that is responsible for centralizing and coordinating all admissions-related processing for the 8 tertiary institutes in Hong Kong.

Graduating high school students can submit a prioritized list of up to 25 different programs (potentially from different universities) to JUPAS for automatic computerized admissions allocation [3].

To produce this list of 25 programs, students have to consider:

- **Unique university requirements**
  Each university may have their own unique set of admission requirements. For example, an applicant may need to satisfy certain language requirements in one university but not in another.
- **Unique program requirements**
  Each bachelor degree program within even the same university may have additional and different requirements. For example, different programs may require qualification in related subjects of study.
- **University ranking**
  Students may have perceived the ranking of the university based on reputation of the universities from various sources.
- **Interest in fields of study**
  Students will also need to consider which bachelor programs will best suit their interests and abilities.

To do this selection manually, students have to laboriously search and study numerous web pages of every university, every department, and every bachelor degree program as well as perform research on websites related to university rankings. This task is extremely time-consuming and may lead to days of online research through over a hundred web pages before a student can produce the required prioritized list of 25 programs that he/she thinks is best for him/her to study as well as will give him/her the best chance of being accepted.

Our "My First Uni" (MFU) application is a research experiment to investigate whether Semantic Web can be used to help streamline this program selection process.

Before we explain our approach, we first describe how one might approach this problem without Semantic Web.

3. Alternative 1 – Centralized Database

One way to solve the Hong Kong admissions problem is to use a centralized database that collects all information related to universities, departments, programs and their admission requirements. This information will need to be collected and consolidated by a third party, a service provider, who will manage this centralized database as well as develop Web applications to give advice to high school students.

The problem with this approach is in the constant need to ensure all information is up-to-date and accurate. This can either be done manually, which will be too labor intensive and error prone, or through sophisticated middleware and integration technology to integrate data from the 8 universities, which will be quite impossible as most of the data reside as free-form textual content.

4. Alternative 2 – XML

An alternative is to use XML for data exchange. If all universities provide admission requirements information in XML form, the application program can easily process information directly from the Web rather than retrieving them from a centralized database. In this way, control over the data remains with the information provider and the access of information is opened to everyone.

But there are many limitations. To make this possible all the XML documents and the application program have to agree to use a common XML structure and tags (hence a common XML schema [4]) [5]. This is not practical as it is difficult to enforce all information providers to use the same schema and that it is difficult to define a schema that suits different variations of information from different universities.

Another problem is the discoverability of these XML documents on the web. Say if there is a new university or some existing XML documents change their URIs, the application program will not be able to discover the location of these XML documents for use. It is difficult for today's search engine to discover these specific XML documents on the web. [6]

In addition, the semantics of the XML documents is not defined within themselves. Applications could interpret the same XML document with different semantics (and probably most applications will misinterpret the original semantics).

5. Alternative 3 – RDF & DAML+OIL

The ideal technical features we are looking for (for our MFU application) can be summarized as follows:

- Information on the Web should be machine-processable to allow Web applications to service user questions by manipulating multiple sources of information from the Web and consolidate them to generate an answer.
- Control and maintenance of information should remain with the information owners.
- Access of information should be opened to public.
- Document storing machine-processable information should be given great flexibility in how it is structured.
- These documents should be discoverable and classifiable in the dynamic and opened Web.
- All information should have well-defined semantics resided within itself so that it is inter-processable by any machine process.

Building on top of XML, RDF [7-9] inherits all the benefits of XML but also incorporated clear semantics in the document. Instead of a representation of a syntactic structure, RDF is a syntax-neutral way to represent a model which can be thought as a directed labeled graph model. Every node and arc in the model must have a URI [10] label, which is as critical as the hyperlink of HTML. With the URI label, RDF documents can be linked with other web documents so that web crawlers can traverse through the web to discover these RDF documents. Thus the URI label greatly enhanced the discoverability and the cohesion of RDF documents.

Since RDF is about describing something about something, or describing anything about anything, there is a great flexibility in composing a RDF comparing with

XML. XML document structure is bounded by XML schema, the whole tree structure and all its tag names has to be consistent with the XML schema. But in RDF, the document structure is not bounded; author can use whatever vocabulary in the RDF statements. A RDF document is valid as long as all the RDF statements are complete.

With DAML+OIL [11-16] semantics incorporated, RDF can extend its expressive power of semantics. DAML+OIL gives formal definition to vocabularies used in the RDF documents. Since the DAML+OIL itself is also a RDF document, it is discoverable and accessible by public. This allows any Web applications to interpret any RDF document with correct semantics defined in the corresponding DAML+OIL documents. These formal semantics definitions are very useful that enable intelligent agents to make decisions, such as inference [17] and prediction.

Alternative 3 is the approach we took in designing the MFU application. The following Sections explain the MFU prototype design in two parts – the Semantic Web content and then the application.

6. Designing the Semantic Web Content

The first step is of course to define an appropriate ontology to use to express knowledge of universities, departments, programs, and their admission requirements. Ontology is an explicit specification of a conceptualization. It is a description (like a formal specification of a program) of the concepts and relationships among these concepts [18]. For example, similar to class-object relationship in object-oriented programming, there is a class-instance relationship in ontology: a class in the ontology is a template of concept and instances can be instantiated from the template. On the representation level, our ontology is represented in DAML+OIL and the instances are represented in RDF.

6.1 Ontology Design Strategy

Our ontology was developed following the design strategy from "Ontology Development 101: A Guide to Creating Your First Ontology" [19] which involves the following steps:

(1) Determine domain and scope of ontology
(2) Consider reusing existing ontologies
(3) Enumerate important terms in the ontology
(4) Define the classes and the class hierarchy
(5) Define properties of classes
(6) Define facets of the properties
(7) Create instances

6.2 Determine Domain and Scope of Ontology

In the MFU prototype, the ontology is specific to the concept of university admission, or more specifically, university bachelor degree program admission for Hong Kong secondary school graduates. The ontology provides information about which Bachelor programs a student is eligible to apply for, and then sort them according to the university ranking. To trim down the scope to a manageable size, most of the supplementary concepts were omitted. Only those essential to determining admission eligibility are included in the ontology. For example, since the scope is defined to include only HKCEE and HKALE examination record as admission criteria, other criteria such as international examination record and extra curricular activities of students are not considered in this ontology.

6.3 Consider Reusing Existing Ontologies

In terms of consistency and reusability, ontology should be reused if there are existing ones that fit the purpose. Unfortunately, there are none that deals with university admissions in the DAML+OIL ontology library [20]. Therefore the MFU ontology had to be built from scratch.

Although there is no existing ontology that fits the purpose, there are some ontologies which are about university research and publication. We have based our MFU ontology on these works.

6.4 Enumerate Important Terms

The following is a sample of some of the terms included in the first draft of the MFU ontology:

University	Dept	Bachelor degree	Student	HK High school graduate
HKCEE	HKALE	Email address	Ranking	Ranking type
HKCEE subjects	HKALE subjects	HKCEE grade	HKALE grade	Entry requirement

6.5 Define the classes and the class hierarchy

Figure 1 is a sample of part of the MFU ontology class hierarchy developed using ontology editor *OilEd* [21].

Fig 1. Class Hierarchy of MFU ontology

top	Denoted the root of the ontology
C	Denoted a class of information in the domain
C #1	Denoted a class in the MFU ontology
C #2	Denoted a class defined in ontology "ksl-daml-desc" from Stanford University

For classes marked "#1", they are classes defined in our new MFU ontology. For classes marked "#2", they are super classes being inherited in MFU. These classes are not originally defined in MFU, but in http://www.ksl.stanford.edu/projects/DAML/ksl-daml-desc.daml, which is an existing ontology modeling university publication. In MFU these super classes are added with new properties. It is important to link with the existing ontology wherever possible in order to extend the existing knowledge base rather than having these ontologies isolated from each other.

This class hierarchy shows the superclass-subclass relationship. For instance, "Student" is a subclass of "Person" and "Student" is also a super class of "Hong-Kong-High-School-Graduate".

6.6 Define Properties of Classes and Their Facets

Similar to the Object-Oriented classes, each of the ontology classes has its own properties. There is no "method" since we are composing the N-Triple/RDF Triple, the Subject–Property- Property Value relationship. (or the Subject-predicate-object relationship). Figure 2 illustrates the MFU class design using UML notations [22].

6.7 Properties and Class relationships

The Class relationships are shown in Figure 3. In the diagram, inherited properties from super class are hidden. Only class restrictions (necessary properties) are shown in the class boxes. Optional properties are not shown in the diagram. Properties marked with "#" have cardinality "exactly 1" to the class. Other properties (which relate to other classes) have cardinality as shown at the ends of the linkage lines.

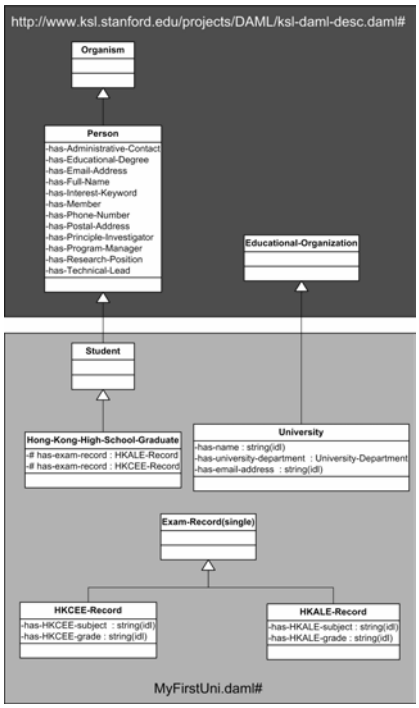

Fig 2. Class Generalization in MFU Ontology

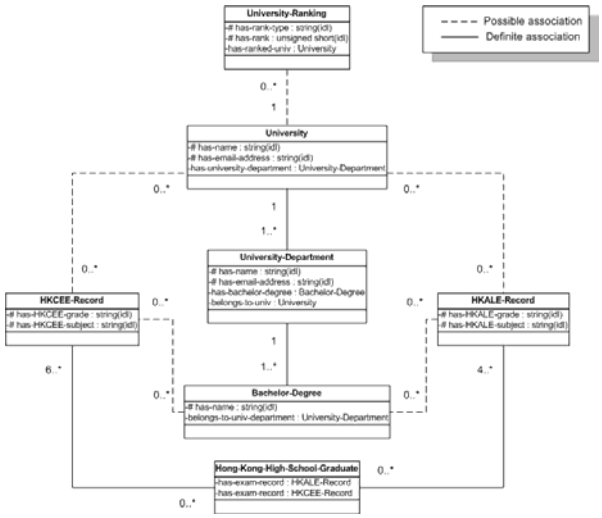

Fig 3. Class Relationship Diagram

Definite Associations:

University-Ranking to University relationship (unidirectional):

Domain	*Property*	*Range*
Class "University-Ranking"	has-ranked-univ	Class "University"

A "University-Ranking" describes the ranking of a "University" in certain area (i.e. the rank-type). A ranking describes one single "University". Yet a "University" may have "University-Ranking", up to many "University-Ranking" as a "University" may have rankings in different areas or ranking by different parties.

University to University-Department relationship (bidirectional):

Domain	*Property*	*Range*
Class "University"	has-university-department	Class "University-Department"
Class "University-Department"	belongs-to-univ	Class "University"

A "University" is defined to have at least one and up to many "University-Department". Each "University-Department" must belong to one single "University" only.

University Department to Bachelor-Degree relationship (bidirectional):

Domain	*Property*	*Range*
Class "University-Department"	has-bachelor-degree	Class "Bachelor-Degree"
Class "Bachelor-Degree"	belongs-to-univ-department	Class "University-Department"

A "University-Department" may offer at least one and up to many "Bachelor-Degree" programs. Each "Bachelor-Degree" must belong to one single "University-Department" only.

Hong-Kong-High-School-Graduate to HKALE-Record relationship (unidirectional):

Domain	*Property*	*Range*
Class "Hong-Kong-High-School-Graduate"	has-exam-record	Class "HKALE-Record"

A "Hong-Kong-High-School-Graduate" is defined as a student who has taken at least 4 subject examinations in HKALE, and thus has at least 4 "HKALE-Record". It is the requirement of some universities in Hong Kong that their local applicant should have taken at least 4 subject examinations in HKALE.

Hong-Kong-High-School-Graduate to HKCEE-Record relationship (unidirectional):

Domain	*Property*	*Range*
Class "Hong-Kong-High-School-Graduate"	has-exam-record	Class "HKCEE-Record"

A "Hong-Kong-High-School-Graduate" is also defined to have taken at least 6 subject examinations in HKCEE, and thus has at least 6 "HKCEE-Record". It is very common that a local university applicant to have taken 6 to 10 subjects in their HKCEE.

Possible Associations:

University to HKALE-Record / HKCEE-Record relationship (unidirectional) :

Domain	*Property*	*Range*
Class "University"	has-entry-requirement	Class "Exam-Record(single)"

It is common that universities in Hong Kong require their local applicants to have certain grade in some of the subjects in their HKALE & HKCEE record. That's why "University" may relate to zero or more "Exam-Record(single)" (which could be a "HKCEE-Record" or "HKALE-record") as the university's admission requirement. It is an optional relation here as to leave some flexibility to the admission requirement.

Bachelor-Degree to HKALE-Record relationship

Domain	*Property*	*Range*
Class "Bachelor-Degree"	has-entry-requirement	Class "Exam-Record(single)"

In addition to the university admission requirements, individual Bachelor degree program may require additional HKALE & HKCEE grades from the applicant who applies for that degree. This is again an optional relation.

6.8 Create instances

After the completion of class and property definitions, the ontology can be used to create instances which are represented in RDF. The following shows two simplified examples out of the many RDF files used in the MFU prototype. The property values of these instances and the mapping between the instances and the RDF documents are shown in tabular format. Each table represents one RDF document.

ranking.rdf

Class	*Instance*	*Property*	*Property value*
University-Ranking	RANKING-OVERALL-BU	has-rank	3
		has-web-page	ranking.htm
		has-rank-type	overall
		has-ranked-univ	BU.rdf#BU
	RANKING-OVERALL-CTU	has-rank	1
		has-web-page	ranking.htm
		has-rank-type	overall
		has-ranked-univ	CTU.rdf#CTU

	RANKING-OVERALL-PLU	has-rank	2
		has-web-page	ranking.htm
		has-rank-type	overall
		has-ranked-univ	PLU.rdf#PLU

BU.rdf			
Class	*Instance*	*Property*	*Property value*
University	BU	has-name	Baptess University
		has-email-address	email@bu.edu.hk
		has-web-page	BU.htm
		has-university-department	BU-CS.rdf#BU-CS
		has-university-department	BU-JN.rdf#BU-JN
		has-entry-requirement	BU.rdf#genid001
HKALE-Record	*genid001	has-HKALE-subject	Chinese
		has-HKALE-grade	E

7. The MFU Application

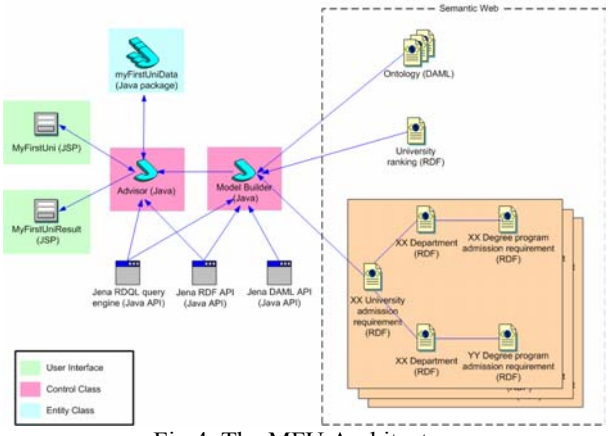

Fig 4. The MFU Architecture

The MFU application uses a typical Web-based architecture using JSP. It has two special control classes:

Advisor – invokes its "Model Builder" to build a RDF model starting from a specified root URI. After the model is ready, it performs various queries to the model for information including the academic fields of the available degree programs and the list of recommended degree programs. It takes user inputs as well as query results and then stores them using corresponding entity classes.

Model Builder –used to retrieve Semantic Web documents (.rdf and .daml) and build a RDF model using retrieved RDF documents [23]. Initially, the Model Builder starts crawling from a root URI given by its "Advisor".

The application also makes use of the Jena Tool:

Jena RDF API [24] – a Java library that supports retrieval, parsing and model manipulation for RDF. It implements the RDF model as a set of RDF Triples and provides resource centric methods for manipulating an RDF model as a set of resources with properties.

Jena RDQL Engine [25] – like SQL, RDQL is a query language for RDF in Jena models. This RDQL engine matches the RDF statement pattern specified in a RDQL query against the directed graph in the RDF model to yield a set of matches.

Jena DAML API – library to support a special RDF model for DAML. This model provides accessor methods to get DAML classes and properties in the model and it is also supported by the RDQL.

MFU firstly allows a student user to enter his/her academic qualification through a simple to use interface:

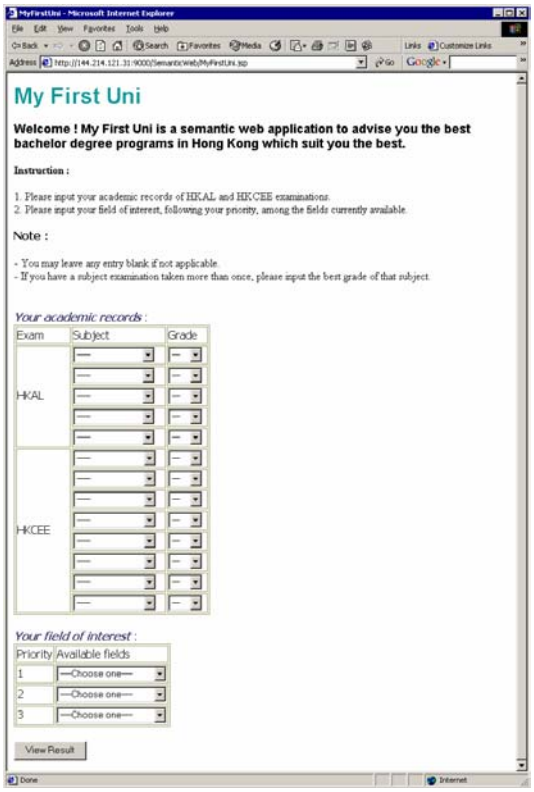

Fig 5. MFU Input Screen

After searching, processing and prioritizing with special matching formula, a resulting list of programs that a student might be interested in and is qualified to apply for will be listed:

Fig 6. Sample MFU Output Screen

8. Search Algorithm for Model Builder

To start building a RDF model, the Model Builder has to retrieve RDF documents from the web. To do this, a root URI has to be assigned to the Model Builder so that it can explore for other RDF documents linked directly/indirectly starting from the root URI. This technique is commonly used by most of the web crawlers today. The gist for effective web-crawling, however, is not about how the crawling starts, but how the crawler explores the links once it is started.

The "explore" action by the Model Builder, which is a search process for RDF documents, has two implementations "Explore by depth" and "Explore by class".

8.1 Explore by Depth

The first implementation "explore by depth" is a traditional bounded depth first search. It explores all the links to Semantic Web documents (.rdf and .daml files) blindly from the root URI up to depth N, (N is a method argument). This implementation, though it is not intelligent, guarantee all documents within the bound are retrieved. After the search, two models are built in the memory, one for all retrieved RDF and the other for all retrieved DAML.

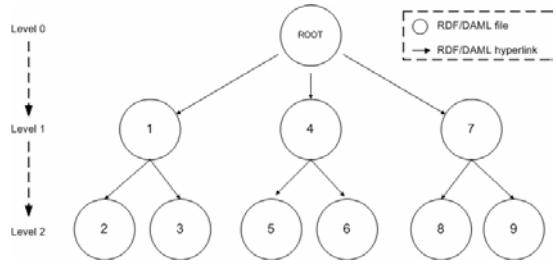

Fig7. Explore by depth up to depth 2

8.2 Explore by class

The second implementation "explore by class" is a new intelligent approach building on top of the bounded depth first search. It is like a search that maximizes the number of retrieval of RDF resources of a specific class. To use "explore by class", a URI of a wanted class and a search depth have to be supplied as arguments.

The basic idea can be explained by an analogy of treasure hunting. Say a treasure hunter wants to hunt for ancient treasures. He can do it by either searching for treasure directly or searching for a map showing the location of the treasure. He starts his search at the centre of a circular boundary and search through everywhere within the boundary. When finished, he has found some treasures and some maps pointing to other treasures out of the searched boundary. The treasure hunter then follows those maps to hunt those out-of-the-boundary treasures by traveling a known distance given from the maps.

The "explore by class" search is similar to treasure hunting, with the search depth as the circular boundary, the instances (RDF) of the wanted class as treasures and the ontologies (DAML) as maps. This search algorithm involves three phases:

In the first phase, the ontology of the wanted class (which is the XML namespace URI [26-27] of that class) is retrieved and stored in a DAML pool. Then, a bounded depth first search is performed up to the specified search depth. In this phase all RDF and DAML documents within the bound (plus the ontology of the wanted class) are retrieved and stored in the RDF pool and the DAML pool respectively.

In the second phase, an associated-class search is performed. All the "concepts" in the retrieved ontologies (DAML documents) are analyzed to look for cardinality constraints (min/exactly equal to N, where N>=1) of any

property in class restrictions, which the range of the property is the wanted class.

The term "cardinality constraint" is the number of occurrence of a property. For example, one may define in an ontology a cardinality constraint for property "has-age", that "If anything has age, that thing has exactly one age". The formal expression for this in Description Logic [28-29] is:

- =1 has-age

The term "class restriction" refers to one or more conditions that are always true for a class. For example, one may define in an ontology a class restriction to the class "human couple", that "for any human couple, if it has a child, the child must be a human". This is always true for all child of any human couple that they must be human but not alien! The formal expression for this in Description Logic is:

- human couple $\subseteq$ $\forall$ has-child.human

The term "range of a property" refers to the object in a subject-predicate-object triple. For example, one may define in an ontology a range for the property "has-age", that the range of "has-age" must be of class "number". The formal expression for this in Description Logic is:

- $\forall$ has-age.number

So the search is looking for a set of associated-class A, where
$A = A_1 \cup A_2$,
$A_1 \subseteq (= N\ ANY.C)$,
$A_2 \subseteq (\geq N\ ANY.C)$,
$N \geq 1$,
ANY = any properties
C = wanted class (for association level 1 search)

After the association level 1 search (the search for direct associated classes), an iterative operation of the same search will be performed. For every associated-class *a* in Set A obtained from the search above, the associated-class search is done again with wanted class C equals to *a*. This is the association level 2 search. It looks for all associated-class of an associated-class of the wanted class. In such way, this iterative search goes on up to level n, where there is no more associated-class of any associated-class found from the ontologies in the DAML pool.

At the end of the second phase, the Model Builder gets the maps. The cost to find a RDF resource of the wanted class starting from a RDF resource of any associated-class is the association level number of that associated-class. Say if the wanted class C has an associated-class Y, and Y has an associated-class Z. Then from any RDF resource of class Z, the cost to find a RDF resource of class C is exactly equals to Z's association level, which is 2.

In the third phase, after a bunch of maps are processed in the second phase, the Model Builder starts to retrieve "out-of-the-boundary" RDF documents with a known cost. It searches for all RDF resources of any associated-class in the set of leave nodes in its search tree, and explore it with the known cost if such resource is found in the leave node set.

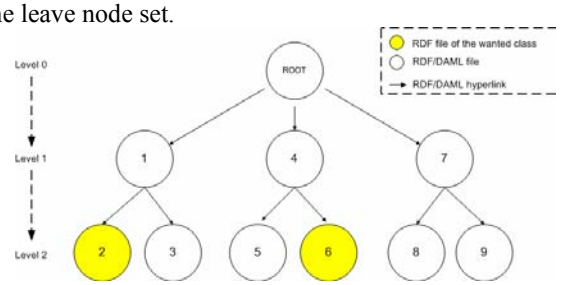

Fig 8. Explore by class : first phase

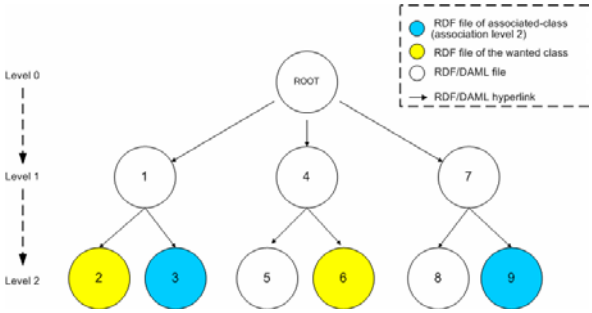

Fig9. Explore by class : post-second phase

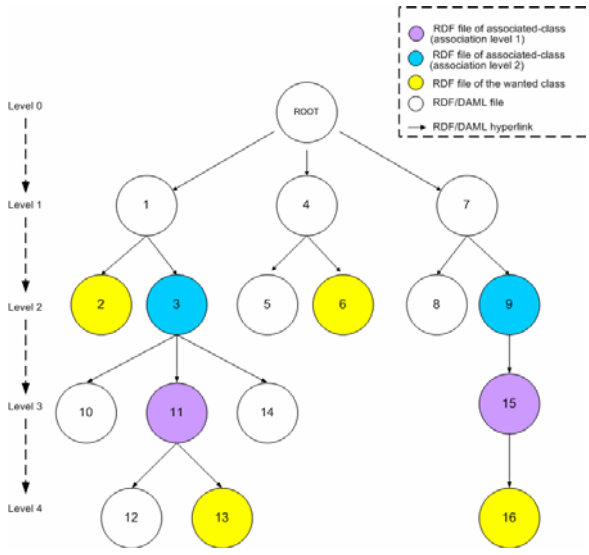

Fig 10. Explore by class : third phase

There are two assumptions that govern the effectiveness of this search algorithm:

(1) In an ontology, some classes have necessary association with other classes by mean of class restrictions (Some classes have associated classes)
(2) Instances (RDF documents) must obey the definitions in its ontology (DAML documents)

If (1) is not satisfied, this "explore by class" search is just the same as a bounded depth first search. The more the associated classes exist, the better the effectiveness of this search. If (2) is not satisfied, this new search is even worse than a bounded depth first search since extra cost is paid without getting any wanted documents.

9. Experiments

The MFU prototype allowed us to experiment with the usage and limitations of our ontology and RDF files under different usage scenarios. For example, in:
(1) Adaptation to newly available resources
(2) Anyone can say anything about anything anywhere
(3) Intelligent web crawling
(4) Basic inference [30]

The experiments allowed us to extend and refine our ontology. The main intension is that this ontology might possibly be shared among the universities in Hong Kong.

10. Conclusion

Semantic Web is, in no doubt, the future of World Wide Web. In this project, we have shown, using a workable MFU prototype, that Semantic Web enables us to use the Web as a world wide data and knowledge base for machine processing. It is a feasible, beneficial, on-demand technology that enables automation of information processing and intelligent services on web that were not possible before. It is an evolution of the Web where the minimalist design or least power principle is conserved: "make the simple things simple, and the complex things possible" [31].

With current state-of-the art in Semantic Web, we can integrate and process distributed heterogeneous web data using RDF and define formal semantics of concepts used in the RDF with DAML+OIL so that software agents can reason about these concepts. For the Semantic Web of tomorrow, we expect a web of trust and proof [32] can be made possible. However Semantic Web is regarded as a vision. It is not something that can be totally done in the future because the requirement for machine processing on web information is also moving forward while we are advancing our web technology. Therefore, Semantic Web will continuously evolve beyond our expectation [33].

10. Acknowledgement

The work described in this paper was substantially supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. 9040517, CityU 1109/00E). This work was also partially supported by a grant from City University of Hong Kong (Project No. 7001286).

11. References

[1] Tim Berners-Lee, Mark Fischetti, Michael L. Dertouzos, *Weaving the Web The original Design and Ultimate Destiny of the World Wide Web by its Inventor,* Harper SanFrancisco, September 22, 1999, ISBN: B00006B5XJ
[2] Tim Berner-Lee, *Semantic Web Road map,* http://www.w3.org/DesignIssues/Semantic.html
[3] JUPAS Hong Kong, *JUPAS Programme Selection,* http://www.jupas.edu.hk/jupas/content_note_bs.htm#programme
[4] W3C, *XML Schema,* http://www.w3c.org/XML/Schema
[5] Semaview, *XML and RDF Illustrated*, http://www.semaview.com/d/RDFandXML.pdf
[6] Norman Walsh, *What is XML?*, October 03, 1998 http://www.xml.com/pub/a/98/10/guide1.html#AEN58
[7] W3C Recommendation 22 February 1999, *Resource Description Framework (RDF) Model and Syntax Specification*, http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/
[8] Tim Bray, *What is RDF?,* http://www.xml.com/pub/a/2001/01/24/rdf.html
[9] Renato Iannella , *An Idiot's Guide to the Resource Description Framework,* http://archive.dstc.edu.au/RDU/reports/RDF-Idiot/
[10] World Wide Web Consortium, *Naming and Addressing: URIs, URLs, ...* , http://www.w3.org/Addressing/
[11] Ian Horrocks, *DAML+OIL: an Ontology Language for the Semantic Web,* http://www.cs.man.ac.uk/~horrocks/Slides/daml-pi-feb-01.pdf
[12] Deborah McGuinness, *DAML-ONT and OIL,* http://www.daml.org/2000/10/daml-oil
[13] Roxane Ouellet, Uche Ogbuji, *XML.com: Introduction to DAML: Part I,* http://www.xml.com/pub/a/2002/01/30/daml1.html

[14] Roxane Ouellet, Uche Ogbuji, *XML.com: Introduction to DAML: Part II,* http://www.xml.com/pub/a/2002/03/13/daml.html
[15] Roxane Ouellet, Uche Ogbuji, *XML.com: DAML Reference,* http://www.xml.com/pub/a/2002/05/01/damlref.html
[16] Daml.org, *Annotated DAML+OIL (March 2001) Ontology Markup,* http://www.daml.org/2001/03/daml+oil-walkthru
[17] The Semantic Web Community Portal, *Inference Engines for the Semantic Web*, http://www.semanticweb.org/inference.html
[18] Erol Bozsak, Marc Ehrig, Siegfried Handschuh, Andreas Hotho, Alexander Maedche, Boris Motik, Daniel Oberle, Christoph Schmitz, Steffen Staab, Ljiljana Stojanovic, Nenad Stojanovic, Rudi Studer, Gerd Stumme, York Sure, Julien Tane, Raphael Volz, Valentin Zacharias, *KAON - Towards a large scale Semantic Web*, http://www.aifb.uni-karlsruhe.de/~sst/Research/Publications/dexa2002.pdf
[19] Natalya F. Noy, Deborah L. McGuinness, *Ontology Development 101: A Guide to Creating Your First Ontology*, http://protege.stanford.edu/publications/ontology_development/ontology101.html
[20] DAML.org, *DAML Ontology Library,* http://www.daml.org/ontologies/
[21] Sean Bechhofer, *OilEd 3.4 Manual*, http://oiled.man.ac.uk/docs/Manual.pdf
[22] Rob Pooley, Perdita Stevens, *Using UML: Software Engineering with Objects and Components,* Addison Wesley, 1999, ISBN: 0201360675
[23] Ontobroker, *Usage of the RDF Crawler of 2000-11-27,* http://ontobroker.semanticweb.org/rdfcrawl/help/
[24] HP Labs, *Jena Semantic Web Toolkit*, http://www.hpl.hp.com/semweb/jena.htm
[25] HP Labs, *RDQL - RDF Data Query Language,* http://www.hpl.hp.com/semweb/rdql.htm
[26] World Wide Web Consortium 14-January-1999, *Namespaces in XML*, http://www.w3.org/TR/1999/REC-xml-names-19990114/
[27] Ronald Bourret, *XML.com: Namespace Myths Exploded,* http://www.xml.com/pub/2000/03/08/namespaces/
[28] KRSS group, Description-Logic Knowledge Representation System Specification, http://dl.kr.org/krss-spec.ps
[29] Ian Horrocks, *Reasoning with Expressive Description Logics: Theory and Practice,* http://www.cs.man.ac.uk/~horrocks/Slides/leipzig.pdf
[30] Sean Bechhofer, Ian Horrocks, Carole Goble, Robert Stevens, *OilEd: A Reason-able Ontology Editor for the Semantic Web*, http://potato.cs.man.ac.uk/slides/oiled-ki2001.ppt
[31] Tim Berners-Lee, Eric Miller, *The Semantic Web*, http://www.w3.org/Talks/2002/01/10-video/
[32] Tim Berner-Lee, World Wide Web Consortium, *Semantic Web - XML2000* http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html
[33] Bhavani Thuraisingham, Bhavani Thuraisingha, *XML Databases and the Semantic Web*, CRC Press, 1st edition, March 27, 2002, ISBN: 0849310318