Statutes on the Semantic Web

Ciaran Mandal

B.A. (Mod) CSLL

Final Year Project May 2005

Supervisor: Dr. Tim Fernando

Declaration

I hereby declare that this thesis is entirely my own work and has not been submitted as an exercise for a degree at any other university.

_____ May 2, 2005

Ciaran Mandal

Abstract

This project is based around research conducted into the continually developing semantic web and its possible applications in the area of legal research, specifically applied to the Statutes of Trinity College, Dublin. The intention is to find a way to more effectively present such documents so as to make them more accessible to people with legal backgrounds and laymen alike.

Acknowledgements

Firstly I would like to thank Tim Fernando for his support and guidance through thick and thin throughout the course of this year.
I would like to thank Carl Vogel for his guidance and saintly patience during these last four years.
I would also like to thank Mary Sharpe and Eithne Healy for helping me out of so many sticky situations in the past!
And finally a heartfelt thanks to my family, whose support has been unwavering and who have always made going home the most satisfying part of the day.

“One's first step in wisdom is to question everything - and one's last is to come to terms with everything”

Georg Christoph Lichtenberg **(1742 - 1799)**

List of Figures

Contents

Introduction

Since the advent of the World Wide Web (WWW) and its relentless advance to becoming a core part of the daily lives of millions of people around the world, for work and for leisure, the way in which information is transmitted, stored, and accessed has been revolutionised. No longer is it necessary to wade through endless stacks of books and library forms to find a given article or essay. Online catalogues, webpages and search engines have reduced the location of articles to a process that begins and ends in the blink of an eye, putting hundreds of thousands of resources at the disposal of anyone connected at the touch of a button.

This material is presented on our screens in countless different ways – frames and windows and pictures all designed to catch our attention and aid our understanding, all made possible by technological advances such as broadband and bluetooth allowing us to view more information at greater speeds and with more freedom of movement. While bookshelves and basement archives will never be totally replaced, there can be no doubt that as a source of information and an aid to learning the WWW is becoming invaluable at all levels of education. Indeed, we are only still coming to terms with the possibilities in areas such as language learning and have yet to settle on the best ways to utilise this technology.

There is a sizeable obstacle holding back the web as we currently know it, however. The information on the WWW is understandable to humans, of course, and can be manipulated and remodelled in many different ways, to aid human understanding. To a machine, though, the material from web pages and articles from the whole of the web, be it a quantum physics paper or an McDonalds menu, is represented in the same way – a sequence of symbols, an effectively meaningless stream,

incomprehensible and more importantly impossible to manipulate automatically.

For example, a google search for a photograph of John Butcher might come up with the results shown in figure 1 below, based on matching the keywords to words contained in various web pages on the internet.

This approach of matching keywords can be useful for certain enquiries, and using constraints such as “” quote marks for example to match whole phrases as opposed to keywords, but is still limited by its very design. Of 118,700 search results below, only 1 would be useful. Computers can do the work of trawling through reams of text to find a given phrase, but currently they cannot “understand” inputs, that is attach a meaning to a word.

The semantic web is a movement towards this machine understanding of semantic concepts inherent in any given human language.

Using a mark-up language, such as XMLS and more recently OWL, content of web pages can now be ‘tagged’ with semantic markers so as to notify the computer as to the meaning or significance of these phrases.

(Fig 1: A google search result after entering the keywords “photograph John Butcher”)

Web Results **1** - **10** of about **18,700** for **photograph John Butcher**. (**0.29** seconds) Tip: Looking for pictures? Try Google Images Solomon D. **Butcher** - Photographs of the Nebraska Homestead **...**
... was staged by photographer Solomon D. **Butcher** to illustrate **...** over the years who published this **photograph** as the **...** The **John** Curry houseThe **John** Curry house, near **...**
www.nebraskahistory.org/lib-arch/ research/photos/thumbs.htm - 10k - Cached - Similar pages

Photograph Collections, Solomon D. **Butcher** Collection
... Read **John** E. Carter's book, Solomon D. **Butcher**, Photographing the **...** photo #48fsThe **photograph** depicted on the cover of the book is the James Pierce home. **...**
www.nebraskahistory.org/lib-arch/research/ photos/highlite/**butcher**/ - 7k - Cached - Similar pages
[More results from www.nebraskahistory.org]

Prairie Settlement: About the **Butcher** **Photograph** Collection
... Between 1886 and 1911 **Butcher** continued to **photograph**. **...** is taken from Solomon D. **Butcher**: Photographing the American Dream by **John** Carter, published by the **...**
memory.loc.gov/ammem/award98/ nbhihtml/about**butcher**.html - 11k - Cached - Similar pages

History For Sale - Olympic, Auto Racing and Other Autographs
... **JOHN** DONALD 'DON' BUDGE - **PHOTOGRAPH** SIGNED - DOCUMENT **...** BUDGE - PRINTED CARD SIGNED, **JOHN** DONALD 'DON **...** Autographs: SUSAN **BUTCHER** - **PHOTOGRAPH** SIGNED, SUSAN **BUTCHER** **...**
www.historyforsale.com/ html/display.asp?page=607&start=3&sort=&signer=&dpp=15 - 39k - Cached - Similar pages

Squidco: Burger, Rob: Lost **Photograph**
(Label: Tzadik)
... Black Peter Blegvad Jaap Blonk **John** **Butcher** Euguene Chadbourne **...** Tétreault Robert Wyatt Otomo Yoshihide **John** Zorn ADD **...** Burger, Rob: Lost **Photograph** (Label: Tzadik **...**
www.squidco.com/ miva/merchant.mv?Screen=PROD&Store_Code=S&Product_Code=1343&Category_Code=TZ - 44k - Cached - Similar pages

Alive & Shouting - The Official Oysterband Discography Website
... Chopper / Lee Partis / Ian Telfer / **John** Jones / Alan Prosser Publicity **Photograph** by Bleddyn **Butcher** circa "Shouting End Of Life" [1995], Lee Partis / Chopper **...**
www.aliveandshouting.fsnet.co.uk/**photograph**s.htm - 8k - Cached - Similar pages

Big Cypress Gallery Clyde and Niki **Butcher** Gift Shop - Books **...**
... **photograph** of Cayo Costa Island #2. The **photograph** and the **...** three photographs written by Tom Schroder and **John** Barry hardbound. **...** The story of how **Butcher** came to **...**
www.clyde**butcher**.com/gifts2.htm - 16k - Cached - Similar pages

Photograph Index
Photograph Index. This is a master index of all the NSW photos on my pages. **...** **John** Wheeler. **...** The tracks are visible as they cross Park Ave. Duncan **Butcher**. **...**
www.triode.net.au/~rolfeb/nsw/allphotos.php3 - 101k - Cached - Similar pages

Custom Puzzle Craft - Wooden Jigsaw Puzzle 107 - **Butcher** on Break
... an already strong image, making it even more successful." **Photograph** sponsored by **...** Name. **Butcher** on Break. **...** **John** S. Stokes III Custom Puzzle Craft 3645 Seventh Ave **...**
www.custompuzzlecraft.com/Evolve/puzzle107.html - 9k - Cached - Similar pages

The Jazz Site: Jazz Services Musicians Database - **Butcher**, **John**
... Bands: Solo (1); Secret Measure (2); **John** **Butcher** Quartet (4). **...** Involved with numerous groups, including **John** Stevens' SME, Chris **...** **Photograph** by Susan O'Connor. **...**
www.jazzservices.org.uk/mus/3190.htm - 3k - Cached - Similar pages

In this project, the Statutes of Trinity College Dublin will be used as a case study to see what (if any!) advantage can be had from converting a plain text file containing the said statutes into html with hyperlinks initially, and then progressing to an OWL file, also using the Protégé 2000 ontology builder, a product of Stanford University research.

The Semantic Web

WWW and W3C

In 1994 at the Massachusetts Institute of Technology (MIT), Tim Berners-Lee, the man credited with the creation of the world wide web, founded the World Wide Web Consortium (W3C) with the aim of "leading the technical evolution of the web".

"In just over seven years, W3C has developed more than fifty technical specifications for the Web's infrastructure. However, the Web is still young and there is still a lot of work to do, especially as computers, telecommunications, and multimedia technologies converge. To meet the growing expectations of users and the increasing power of machines, W3C is already laying the foundations for the next generation of the Web. W3C's technologies will help make the Web a robust, scalable, and adaptive infrastructure for a world of information."
(from WC3 Mission Statement, www.w3c.org)

In brief, the W3C's long term goals are:

- Universal Access
- Semantic Web
- Web of Trust

The W3C's role is that of an umbrella over the whole development of the web. To ensure greater accessibility while striving to standardise the almost daily advances in hardware and software, the W3C tries to ensure that all the developers, programmers, users, and everyone else involved in

the evolution of the web is pulling in the same direction. It does this primarily by publishing recommendations for standards to be adopted. Given that among the 359 members (as of 1/4/04) are such influential IT companies as Microsoft, Intel, AOL, Apple, Ericsson, Nokia, Sun and Sony, it is reasonable to say that what the W3C recommends today becomes standard on your PC tomorrow.

For this reason their second long term goal is important in terms of this project.

The semantic web is not a remote possibility of how the web might evolve; it is rather a certainty that in ten or so year's time it will be the standard, having replaced the 'old-fashioned' html web. It is therefore important to start finding out how best to implement this semantic web in relation to legal documents, to see if and how the semantic web technology can be utilised to aid understanding and research of these resources.

The W3C co-ordinates efforts from a multitude of separate projects and undertakings to help today's web evolve into tomorrow's semantic web.

HTML

Hypertext Mark-up Language, or HTML, is the current lingua franca for publications on the World Wide Web. It is, as the name suggests, a mark-up language based on SGML. It uses tags to modify different types of text and formatting – and opening tag, e.g. <hl>, and a closing tag, </hl>.
It has been reformulated over the last few years to XHTML allowing access from a wide range of browser platforms, such as mobile phones, cars, palm pilots, and so on.

```
<html><head>
                    <title>Ciaran Mandal</title>
          </head><STYLE>
                    BODY{background-color:green;
                              font-size:20pt;}</STYLE>
<body>              <TABLE Border=4>
<TR><TD> Ciaran Mandal </TD>
                              </TR>
                              <TR><TD> SS Computer Science, Linguistics and French </TD>
          </TR>
                              <TR><TD> Trinity College, Dublin </TD>
                              </TR>
                              <TR><TD>  <A HREF="statutes.htm">Linkto the College
Statutes</A></TD>     </TR>
                    </TABLE>
mandalc@tcd.ie<br/>
Final Year Project: Statutes on the Semantic Web<br/>
Supervisor: Dr Tim Fernando<br/>          </body>
</html>
```

(Fig. 2: Example HTML text)

XML

Extensible Mark-up Language (XML) is a mark-up language for documents containing structured information. This information concerns not only content (i.e. words, pictures etc.) but also a marker as to the function of the content. For example, content in a heading can have a different function to that of a footnote, which again could be different from the content of a caption…).

The XML specification defines a standard way to add mark-up to identify the structures inherent in almost every document.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <!--
 Edited with XML Spy v4.2
 -->
- <breakfast_menu>
- <food>
  <name>Belgian Waffles</name>
  <price>$5.95</price>
  <description>two of our famous Belgian Waffles with plenty of real maple syrup</description>
  <calories>650</calories>
    </food>
- <food>
  <name>Strawberry Belgian Waffles</name>
  <price>$7.95</price>
  <description>light Belgian waffles covered with strawberries and whipped cream</description>
  <calories>900</calories>
    </food>
- <food>
  <name>Berry-Berry Belgian Waffles</name>
  <price>$8.95</price>
  <description>light Belgian waffles covered with an assortment of fresh berries and whipped
     cream</description>
  <calories>900</calories>
    </food>
- <food>
  <name>French Toast</name>
  <price>$4.50</price>
  <description>thick slices made from our homemade sourdough bread</description>
  <calories>600</calories>
    </food>
- <food>
  <name>Homestyle Breakfast</name>
  <price>$6.95</price>
  <description>two eggs, bacon or sausage, toast, and our ever-popular hash
     browns</description>
  <calories>950</calories>
    </food>
    </breakfast_menu>
```

(Fig. 3: An example food menu in XML)

Fig. 4: XML Development Goals

1. It shall be straightforward to use XML over the Internet. Users must be able to view XML documents as quickly and easily as HTML documents. In practice, this will only be possible when XML browsers are as robust and widely available as HTML browsers, but the principle remains.
2. XML shall support a wide variety of applications. XML should be beneficial to a wide variety of diverse applications: authoring, browsing, content analysis, etc. Although the initial focus is on serving structured documents over the web, it is not meant to narrowly define XML.
3. XML shall be compatible with SGML. Most of the people involved in the XML effort come from organizations that have a large, in some cases staggering, amount of material in SGML. XML was designed pragmatically, to be compatible with existing standards while solving the relatively new problem of sending richly structured documents over the web.
4. It shall be easy to write programs that process XML documents. The colloquial way of expressing this goal while the spec was being developed was that it ought to take about two weeks for a competent computer science graduate student to build a program that can process XML documents.
5. The number of optional features in XML is to be kept to an absolute minimum, ideally zero. Optional features inevitably raise compatibility problems when users want to share documents and sometimes lead to confusion and frustration.
6. XML documents should be human-legible and reasonably clear. If you don't have an XML browser and you've received a hunk of XML from somewhere, you ought to be able to look at it in your favourite text editor and actually figure out what the content means.
7. The XML design should be prepared quickly. Standards efforts are notoriously slow. XML was needed immediately and was developed as quickly as possible.
8. The design of XML shall be formal and concise. In many ways a corollary to rule 4, it essentially means that XML must be expressed in EBNF and must be amenable to modern compiler tools and techniques.
   There are a number of technical reasons why the SGML grammar cannot be expressed in EBNF. Writing a proper SGML parser requires handling a variety of rarely used and difficult to parse language features. XML does not.
9. XML documents shall be easy to create. Although there will eventually be sophisticated editors to create and edit XML content, they won't appear immediately. In the interim, it must be possible to create XML documents in other ways: directly in a text editor, with simple shell and Perl scripts, etc.
10. Terseness in XML markup is of minimal importance. Several SGML language features were designed to minimize the amount of typing required to manually key in SGML documents. These features are not supported in XML. From an abstract point of view, these documents are indistinguishable from their more fully specified forms, but supporting these features adds a considerable burden to the SGML parser (or the person writing it, anyway). In addition, most modern editors offer better facilities to define shortcuts when entering text.

(from www.xml.com)

XML can ‘tag’ content so a page that the machine sees as:

∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠ
〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□
□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ
∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβ
EB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖ
ϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB
□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□
ΦΙ∃AB□
□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ
∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠ
φ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB
∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□
αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□
∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠ
φ〉□ΦΙ

becomes:

<name>∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉</name>
<location>□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ
□αβEB∏□ΘΠφ〉□ΦΙ∃AB□</location>
<date>□ωϖϕ□αβEB∏□Θ<date>
<location>Πφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□
ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβ</location>
<participants>EB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉
□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□
ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβ
EB∏□ΘΠφ〉□ΦΙ∃AB□
□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ
∃AB□□ωϖϕ□αβEB∏□</participants>
<introduction>ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃
AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ
□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□α
βEB∏□ΘΠφ〉□ΦΙ∃AB</introduction>
<speaker>□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□
ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβEB∏□ΘΠφ〉□ΦΙ∃AB□□ωϖϕ□αβ
EΘΠφ〉□ΦΙ</speaker>

However, although this is a higher level of sophistication and detail thatn html, the machine still sees the same thing, only this time with tags, <ΘΠφ⟩□> and </ΘΠφ⟩□>, just as meaningless as the content that they contain.

There is therefore still a need to add semantics to these tags. There have been many suggestions as to how to go about doing this. An external agreement, between users, could be devised as to the meaning of these annotations, and thus standardising them to such an extent that it wouldn't matter that the machines can't understand them, there would be no need as they would be universal. In other words, some co-ordinating organisation, such as DAML, could agree a meaning for a set of annotation tags that could then be used by everyone. This would be useful for exchanging information between parties who have agreed to the meaning of terms beforehand.

This approach has its drawbacks, however, in that it is very rigid and inflexible, and only a limited number of things can be expressed. It is merely a means of papering over the cracks in an outdated system. What was needed was an overhaul of the very principles behind electronic publication, hence the emergence of ontologies and ontology-based languages to specify the meaning of these annotations.

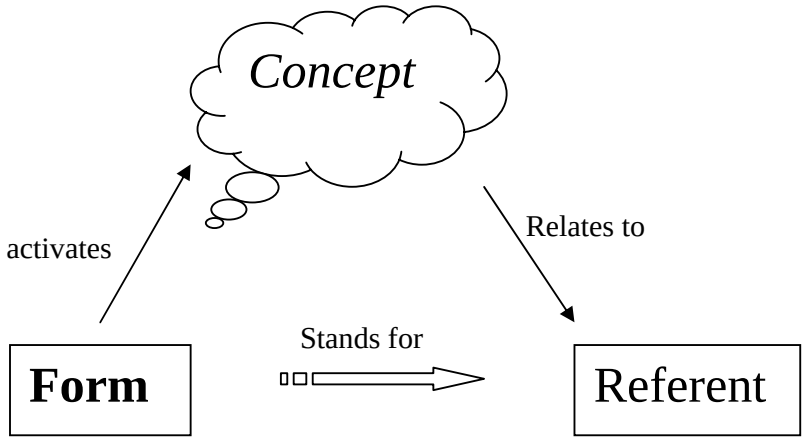
Ontology

The philosophy of ontology tries to answer the questions "What characterises being?" and even "what is being?".

> "*… a philosophical discipline – a branch of philosophy that deals with the nature and the organisation of reality*"
>
> (Science of Being, Aristotle)

In the field of linguistics ontology in its simplest form revelves around 3 ideas – those of the Form, the Concept, and the Referent. The Concept is the idea in one's mind about what is to be referred to. The name of this thing being referred to is the Form. The actual physical thing being referred to is the Referent.

Below is a diagram, the semiotic triangle, from Ogden, Richards, 1923:



(Fig. 5: The Semiotic Triangle)

In terms of Computer Science, according to Horrocks and Sattler,

> *"An ontology is an engineering artefact constituted by a specific vocabulary used to describe a certain reality. A set of explicit assumptions regarding the intended meaning of the vocabulary. Thus an ontology describes a formal specification of a certain domain: shared understanding of a domain and an interest; formal and machine manipulation of a domain of interest."*

Or:

> *"An explicit specification of a conceptualisation"*
>
> (Gruber 1993)

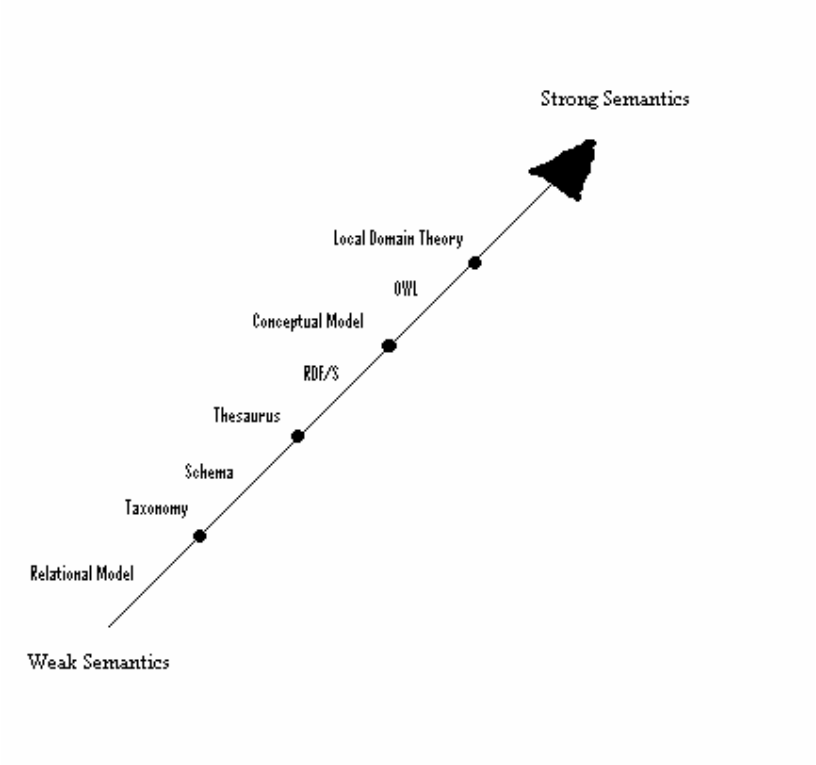
In other words, an ontology describes the meaning of a given concept. For example, an ontology of "Person" could have sub-classes of man, woman, and attributes age, sonOf, daughterOf, fatherOf, and so on.

An ontology can be a taxonomy (knowledge with a parent-child structure), a thesaurus (words and synonyms), a conceptual knowledge (more complex knowledge), or even a logical theory (very rich, complex, consistent and meaningful knowledge).

The Ontology Spectrum

When considering various ontologies, one notion that can be helpful in assessing their effectiveness is that of *semantic richness*. Semantic richness in contrasting ontologies can be thought of in terms of weaker semantics or stronger semantics. At the weaker end of the scale, only very simple meanings can be expressed. At the stronger end, you can express very complex and rich semantics.

There are many different information representations that can be thought of as ontologies, with varying degrees of semantic richness. These are represented in fig. 5 on the Ontology Spectrum.
This scale was devised to compare the semantic richness of classification and knowledge based models. Starting from “weak” at the bottom-left end of the spectrum and rising towards “strong” at the upper-right, it compares the various different types of information representations that can be thought of as ontologies, from a simple taxonomy to a semantically rich and complex local domain theory. It is thus an easy reference point to decide what kind of ontology a given information representation is.



(Fig 5: The Ontology Spectrum)

Taxonomies

At the bottom left of the spectrum, and thus at the semantically weak limit of what can be considered an ontology, is a *Taxonomy*. The creation of taxonomies in machine-usable form is, as the Ontology Spectrum would suggest, a primary step in the development of the semantic web. A taxonomy is a way of classifying or categorising a set of things. This classification is in the form of a hierarchy or tree. It defines a word or idea by demonstrating its relationship with other words or ideas. A user/machine can therefore understand the semantics of a word by reference to its relationship with the words around it in the hierarchy, that is to say the words that it is either i) a sub-classification of or ii) a super-classification of.

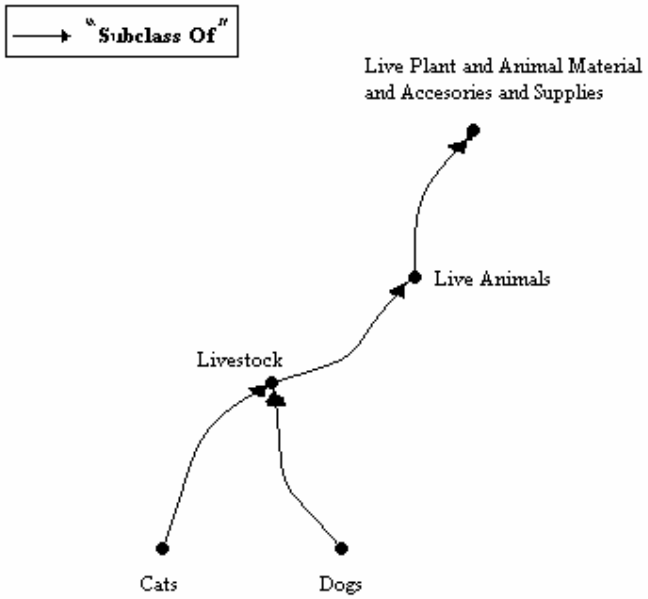
The conventional representation of a taxonomy is shown in fig. 6
It is shown with its root at the top, that is the most general term, with arrows branching down to the more specific terms. The arrows can therefore be taken as meaning “is sub-classification of”. Alternatively, if the arrow is pointing upwards, it can mean “is super-classification of”. These relations can also be called “is subclass of” or “is superclass of”.

Taxonomies are semantically at the weaker end of the spectrum, however they can be made stronger by using the notion of a distinguishing property. This means that each class is made unique by containing at least one property that is different to any other subclasses of its parent class.

A good example from Daconta’s “The Semantic Web” is briefly described here. In a taxonomy of animal species, Mammal and Reptile would both be subclasses of the parent class Vertebrata. Although both

mammals and reptiles have four legs (common properties), mammals are warm-blooded and reptiles are cold-blooded. Therefore warm-bloodedness can be considered one of the properties that distinguishes mammals and reptiles. Another distinguishing property is the property of egg-laying. Although there are exceptions, mammals in general do not lay eggs, but reptiles do.
Using these distinguishing properties all the way down a taxonomy adds to its richness and makes it more expressive.

The next point up and right along the Ontology Spectrum is the Thesaurus.



(Fig. 6: A Sample Taxonomy)

Thesaurus

A thesaurus is a controlled vocabulary that groups together words that are related to each other. There are four different types of relationships that are used in a thesaurus – equivalence, homographic, hierarchical, and associative.

An equivalence relation says that a term X has the same or very similar meaning as a term Y, such as “document” and “file”.

A homographic relation says that X has the same spelling as Y, but a different meaning, such as “tank” – a military vehicle – and “tank” – an object which contains liquids.

Hierarchical relations can be either ParentOf or ChildOf relations. For example, “animal” can be considered a ParentOf “cat”, and similarly “cat” is a ChildOf “animal”. Another way of saying this is that “animal” has a broader meaning than “cat”, so that all cats are animals, but not all animals are cats.

An associative relation says that a term X is associated or has some unspecified relationship with a term Y. For example, if you think of a computer, you can also think of a keyboard, a mouse, a monitor, a hard drive, and so on. Therefore we can consider “keyboard”, “mouse”, “monitor”, and “hard drive” as terms associated with “computer” (note - these terms are all also associated with each other).

A thesaurus ensures consistency in the terms and vocabulary used to describe any given concept.

Logical Theory

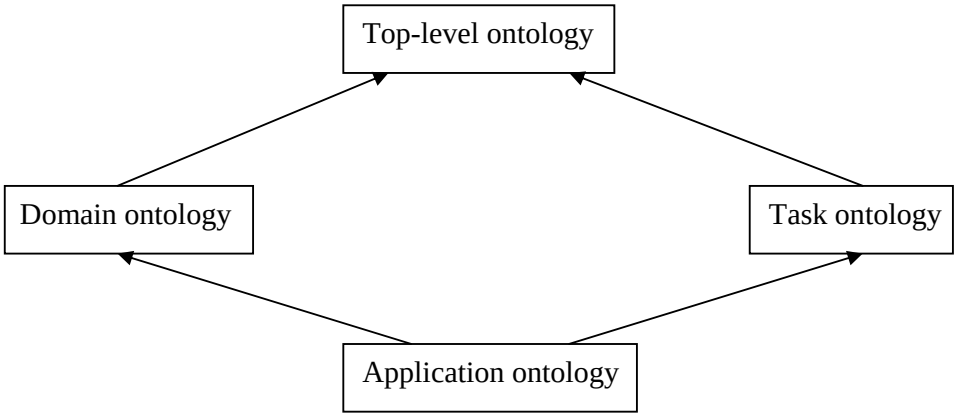
A logical theory is an ontology that is directly semantically interpretable by computer software. To achieve a logical theory an ontology must be built using machine interpretable language, possibly by using an ontology builder program such as the Protégé program used in this project. Using this ontology builder, a logical theory can be constructed and incorporate complex semantic relations and constraints, instances can be creates, and then these instances can be queried by the user, with the computer doing the “hard work” that they have got away without doing for so long!

Topic Maps

Topic Maps are a subject-based classification technique. Topic Maps deal with the idea of classifying documents and section of documents according to their content. The topic map standard arose out of the need to merge indexes of various different documents. The standard is based on the concepts embodied in indexes – Topics, Associations, and Occurences.

A topic can refer to anything whatsoever, a person, an entity, an idea, anything. An occurrence is any resource that deals with the topic in question, such as a webpage or a book. An association is similar to a relation, and describes how separate topics are connected, or associated. For example, if we have 2 topics *Dublin* and *Christchurch*. These could be linked using an association *Is In*, so Christchurch *Is In* Dublin.

Types of Ontologies:



Top-level ontology: describes very general concepts like space, time, event, which are independent of a particular problem or domain. It seems reasonable to have unified top-level ontologies for large communities of users.
Domain ontology: describe the vocabulary related to a generic domain by specialising the concepts introduced in a top-level ontology.
Task ontology: describe the vocabulary related to a generic task or activity by specialising the top-level ontologies.
Application ontology: These are the most specific ontologies. Concepts in application ontologies often correspond to roles played by domain entries while performing a certain activity.

(Carole Goble, Nigel Shadbolt, Ontologies and the grid tutorial. From Dave Lewis, Introduction to Ontology based semantics)

(Fig. 8: Types of Ontologies)

A typical ontology consists of two parts, names and background knowledge. The names are for important concepts in the domain:

- Elephant is a concept whose members are a kind of animal
- Herbivore is a concept whose members are exactly those animals who eat only plants or parts of plants
- Adult_Elephant is a concept whose members are exactly those elephants whose age is greater than 20 years

The background knowledge can include constraints on the domain:

- Adult_Elephants weigh at least 2,000kg
- All elephants are either African_Elephants or Indian_Elephants
- No individual can be both a herbivore or a carnivore

If the light at the end of the tunnel is a world wide semantic web, there are some preliminary advances that must be made in order to ensure this progression. The existing mark-up languages must be extended to provide for semantic constraints, and a common syntax must be agreed upon, so that the semantic web can be based on standards such as HTTP and HTML that the syntactic web is based on today.

As Tim Berners-Lee noted, the challenge of the semantic web is to find a representation language powerful enough to support automated reasoning but simple enough to be usable.

OWL

OWL has been designed to meet the need for a web ontology language.
As detailed before, XML provides a syntax for structured documents but places no semantic constraints on their meaning. XML Schema restricts the structure of XML documents and extends XML with datatypes. RDF is a datamodel for resources and the relations between them. It provides a simple semantics for this model, and these semantics can be represented in an XML syntax. RDF Schema is a vocabulary for describing properties and classes of RDF resources, with a semantics for generalisation-hierarchies of such properties and classes.
OWL adds more vocabulary for describing properties and classes, such as but not limited to relations between classes, cardinality, equality, richer typing of properties, characteristics of properties, and enumerated classes.
There are three OWL sublanguages, OWL Lite, OWL DL, and OWL Full. OWL Lite supports those users primarily needing a classification hierarchy and simple constraints, and is therefore easier to provide tool support for this sublanguage than the more complicated and wide-ranging DL and Full versions. OWL DL is more expressive but still ensures that ensures *completeness*, i.e. all the presuppositions will compute; and *decidability*, that all calculations will in fact terminate. OWL DL is named as it corresponds with description logics, a field of research concerning a particular fragment of decidable fragment of first order logic. OWL Full has maximum expressivity but does not guarantee computation. One major obstacle the will hinder OWL Full becoming a universal standard is that it will probably not be possible to be supported by any reasoning software. That is to say it would not be implemented in

any quick user-friendly construction program or wizard for the uninitiated user as it is too complicated and all-embracing.
Within this framework, the inter-compatibility of these three forms of OWL should be mentioned.

OWL Lite → OWL DL → OWL Full

i.e. Every OWL Lite ontology or conclusion is a legal OWL DL ontology or conclusion, but not the inverse, and so on for OWL DL and OWL Full.

Protégé 2000

Protégé 2000 was developed by Stanford Medical Informatics at the Stanford University School of Medicine. It is an integrated software tool for system developers and domain experts to develop knowledge based systems. A knowledge based computer system includes a knowledge base about a given domain and programs that include rules for processing the knowledge and for solving problems relating to the domain.

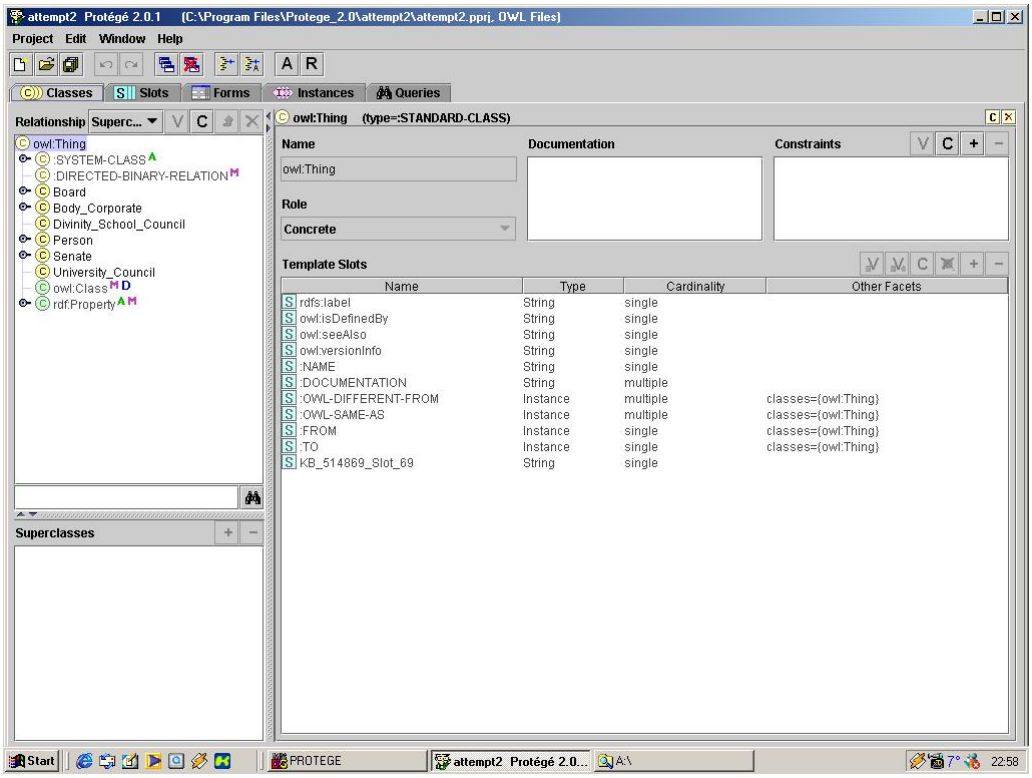
Protégé 2000 consists of a graphical user interface (GUI) whose top-level consists of overlapping tabs for classes, instances, slots, forms, and queries to allow for compact presentation of these parts and convenient editing and 'jumping' between them. This design allows us to:

- model an ontology of classes
- create a knowledge-acquisition tool for collecting knowledge
- enter instances of data and create a knowledge base (KB)
- execute applications

This knowledge base can then be used with a problem-solving method (PSM) – a computer program used with a KB to answer questions or solve problems regarding the domain.

Knowledge-based systems are usually expensive to build and also to maintain. They also usually involve the input of teams of both developers and domain experts who may or may not have experience of working with computers. Protégé 2000 was designed as an aid to both developers and domain experts as a user-friendly interface for creating these KB systems. It is designed to guide them through the process of system development. It also allows users to re-use existing domain ontologies and methods, and so shortens the time needed for development and

maintenance. Many different applications may use the same ontology to solve different problems. Unfortunately after scouring the internet there were no ready-made ontologies out there suited to this project's research! Protégé 2000 was developed originally for, and is mainly used at present in the field of clinical medicine and the biomedical sciences, but it can be used in any area where the concepts can be modelled as a class hierarchy.



(Fig. 9: A screenshot of Protégé 2000)

The development of a successful knowledge-based system built with Protégé-2000 is more of an art than a science. Nonetheless, we can suggest a standard pattern of use that new users should follow to avoid some possible problems of systems development. Protégé-2000 is designed to support *iterative development*, where there are cycles of revision to the ontologies and other components of the knowledge-based system. Therefore developers should not expect to "complete" ontology development without considering other aspects of the process.

For the development of a successful Protégé-2000 project, we would recommend the following steps:

1. Plan for the application and expected uses of the knowledge base. This usually means working with domain experts that have a set of problems that could be solved with knowledge-base technology.
2. Build an initial small ontology of classes and slots, as explained in the Classes and Slots strand.
3. When you have built this ontology (and later when you have extended it or opened it from file), you can directly view forms for entering instance knowledge into the ontology, because Protégé-2000 generates initial forms "on the fly", in its role as a KA-tool generator.
4. You use these forms for acquiring slot values of your test instances, as explained in the Instances strand. At this point, it is usually appropriate to show the ontology and the filled-out instance forms to the domain experts or your expected users. This inevitably leads to a set of revisions, both to the ontology (2.) and to the forms (5.). Note that ontology modifications can be expensive, since some sorts of change could force rebuilding some or all of the knowledge base.
5. Customize the forms to a refined knowledge-acquisition tool, as explained in the Forms strand. While constructing this customized version of the KA-subtool, further design problems in the original ontology may become apparent. If necessary revise the ontology and repeat at 4.
6. With your domain experts, build a somewhat larger knowledge-base that can be tested with your application or problem-solving method.
7. Test the full application with your end-users. This step can lead to further revisions to the ontology and the KA-subtool.

(Fig. 10: Steps for Developing a Protégé Ontology)
From protege.stanford.edu

Statutes

Background

In this project, the statutes used as a basis for research and development using HTML and Protégé to find out the benefits of each are the 1966 Consolidated Statutes of Trinity College Dublin and the University of Dublin, as amended up to and including the Ordnance of 22 March 1999.
They are the foundation for the running of the University. All of the official committees and working posts, educational and administrative, paid and unpaid, elected or honorary, of Trinity College Dublin are provided for somewhere in this document.
The codes of conduct and procedures for dealing with any perceived breach of discipline by both students and staff are contained, as are the procedures for the order of any meetings summoned of the Board, the Senate, the Divinity School Council or any other body of College.
However, like any legal document the language used as well as the convoluted nature of the University's hierarchy and system of elections and subordination makes the statutes difficult to decipher for the lay person trying to enquire about a particular aspect of the statutes. For example, a representative maybe elected to a position by a committee, and one might therefore consider it safe to assume that this person would remain accountable to the committee that elected him, but this is in fact not always the case. For another example, if Mr Jones is subordinate to Mr Smith, and Mr Smith is subordinate to Mrs Murphy, it may not necessarily be the case that Mr Jones is subordinate to Mrs Murphy, such is the complicated nature of the positions provided for in the statutes.
It is therefore difficult for a casual reader to get a grasp of where exactly everyone who holds a post in college 'fits in' to the broader scheme of

things. This is what I hope to clarify using the technology explained in the previous pages.

If these statutes are available to a user as a conventional word or HTML document, there are limited means by which to search or ‘query’ the information. One could do a conventional search, specifying particular keywords and the computer could return all of the matches. However as explained in the introduction, if a user wanted to find out an explanation of the duties of the Provost, for example, a search for the keyword “provost” could return upwards of 100 matches in this document, when only one would be of any use.

Statutes in HTML

One way of taking advantage of HTML tags was to bookmark all of the key sections of the statutes and use each occurrence of keywords as a link back to their definition.

To extend on the 'Provost' example, if each occurrence of the word 'provost' was set-up to be a hyperlink back to the chapter of the statutes that deals with the Provost (Chapter 4), then if a user needed an explanation as to what was being referred to by the word 'provost' he could click on it and be brought to the relevant passage.

This is in fact quite an effective and user-friendly way to present the statutes, and certainly eases a lot of the burden on the user in terms of information management, it becomes easier to navigate one's way through the material. However, this is again simply papering over the cracks. The computer itself is still not processing or manipulating any information, it is the author of the HTML code who must decide what bookmarks to put where and correspond to what, and there is no scope for semantic querying or searching. In other words, the user is still doing most of what Horrocks and Sattler would call 'the hard work", and the computers are still merely presenting the information given to it by the author – the "easy work".

The aspect of the statutes on which this research is concentrated is the hierarchical structure of the various committees (Board, Senate etc.) and the positions within this structure (Chancellor, Fellows, Statutory Officers etc.). This hierarchy is laid out in chapters 2-8, 10-15, 18, and 23 of the statutes.

Design

Using Ontologies to Express Concepts of Statutes

The work in this project concentrates on the specific area of the statutes as outlined above, the power structures, and these structures fit well into a class hierarchy as mentioned above.
Although there is no right or wrong way to model an ontology, in the course of this project, the guidelines followed were those of Noy and McGuinness' Ontology Development 101:

Steps to Create an Ontolology

1) Decide on the domain and scope of the ontology. For this we need to answer some questions:
   a) What is the domain that the ontology will cover? In this case the power structure of the statutes.
   b) For what are we going to use the ontology? To answer queries about these structures, about who answers to who and for what.
   c) For what types of questions the information in the ontology should provide answers? Once instances have been filled in the questions can be anything as simple as: 'Who is the boss of Mr X?' or as complex as 'How many females sit on the board that are over 40 years of age and earning more than €40k in the faculty of Systems Sciences?'.
   d) Who will use and maintain the ontology? The ontology will be used by any student or staff wishing to query the inner workings of

this power-structure ascpect of the statutes, not necessarily very computer-literate, this ontology should be accessible to even the most casual of users. It could be maintained by any Protégé literate user, whose job would be to update the various instances, i.e. when there is a change of Provost or when the new scholars are announced.

A useful technique for determine the scope of an ontology is by devising some **competency questions,** questions that the complete ontology should be able to answer. As well as the examples above in part (c) there could also be questions like:

- Which staff will be reaching 65 years of age in the next 12 months?
- How many scholar positions will be available in 2007?
- When does the current term of Provostship expire?

From the questions above, the ontology will have information of a person's date of birth, faculty, salary, sex, and so on.

These questions are not exhaustive, but can be a useful touchstone during the process to ensure work goes in the right direction.

2) Consider Re-Using Existing Ontologies

There are vast libraries of reusable ontologies available on sites such as www.daml.org, or www.ksl.stanford.edu/software/ontolingua, and it is always worth checking if an existing ontology can be refined to suit the needs of the ontology being developed. Sometimes it is necessary if one's system needs to interact with other applications that have are already using their own ontologies or controlled vocabularies.

In this project's research however, there was no ready-made ontology available at the time of writing, and so the ontology presented in this case was developed from scratch.

3) Enumerate Important terms in the Ontology

   The next step is to write down a list of the terms that are to be either used in the classes or to be explained to the user. What terms would we like to talk about? What are the properties that these terms have? For example, in our ontology we can have terms such as *provost, board, divinity_council, junior_dean, faculty, surname, date_of_birth*. Initially it is useful to get a comprehensive list of terms without worrying about the overlap between concepts or relations among the terms, or whether these terms should be slots or classes in their own right.

4) Define the Classes and the Class Hierarchy.

   There are three widely accepted approaches for developing a class hierarchy.

   - A **top-down** development process defines first the broadest concepts in the domain and then defines in turn more and more specific concepts. For example, we can create an abstract class *Board*, and then the classes that make up the board, *Provost*, *Fellows*, and then the sub-classes *Junior_Fellow*, and *Senior_Fellow*.
   - A **bottom-up** process goes, as the name would suggest, the other way around. Starting with the most specific concepts, *Senior_Master_Non_Regent*, to the superclasses *Regent* and *Non_Regent*, and the subsequent super-class *Master_Arts*.

- A **combination** process is a mixture of the two. Firstly define the most obvious concepts and then both specifying and generalising other classes around these initial ideas.

As noted above about these ontology building methods in general, not one of these three class hierarchy processes are necessary more correct or effective than the other. It depends simply on the developers personal preference with regard to the domain.

Rosch 1978 maintains that the combination approach is often the easiest as the concepts 'in the middle' are usually the more descriptive concepts of the domain.

5) Define the properties of classes – slots

Once we have defined some of the classes, we must describe the internal structure of the concepts. Having already selected classes from the list of terms we created in step 3, the remaining terms form that list will usually be properties of those classes (*date_of_birth*, *faculty*).

A slot should be attached at the level of the most general class that can have that property, as each subclass will inherit this property. For example, *date_of_bith* or *name* should be attached at the most general class *Person*, whereas the *salary* slot can only be attached to the class *Employee* and its subclasses.

6) Define the Facets of the Slots

Each slot is defined by its facets describing data type, allowed values, the number of values (cardinality), and other features a slot can take. For example, the slot *name* (a person's name) is a slot with value type

String, and can only have one value. A slot can have one of a number of different types:

- A **string** data type is used for values such as *name* or *birthplace*, it consists of a set of characters. E.g. “Ciaran” or “This*(&%Is*(A£”%String”
- **Number**, **Float,** or **Integer** types are used for numeric values
- **Boolean** slots contain simple true/false values
- **Enumerated** slots specify precisely what values are allowed in the slot. For example a slot *faculty* may contain one of 6 entries: “Arts (Humanities)”, “Arts (Letters)”, “Business Economic and Social Studies”, “Engineering and Systems Sciences”, “Health Sciences, or Science”. In Protégé 2000 enumerated slots are of type **symbol**
- **Instance** slots allow definition of relationships between individuals. For example a slot *Nominated_By* will have an instance of type *Person* as its value.

7) Create Instances

Finally one must create individual instances of classes in the hierarchy. Defining an individual instance of a class requires choosing a class, creating an individual instance of that class, and filling in specific slot values.

An instance of class *Pro_Chancellor* (it shall be defined in the building of this ontology that there can be no more than 6 instances of *Pro_Chancellor* at any given time, as per chapter II paragraph 3 of the statutes) might have the following slot values:

- Date_Of_Birth: 15/1/1947
- Name: Stephen Knowles

- Faculty: Arts (Humanities)
- Salary: 45000
- Nominated_By: John Phillipson

The Statutes Ontology

- owl:Thing
  - :SYSTEM-CLASS
    - :META-CLASS
      - :CLASS
        - :STANDARD-CLASS
          - owl:Class
        - :OWL-CLASS
          - :OWL-ANONYMOUS-CLASS
            - :OWL-ENUMERATION-CLASS
            - :OWL-RESTRICTION
              - :OWL-ALL-RESTRICTION
              - :OWL-HAS-RESTRICTION
              - :OWL-MAXCARDI-RESTRICTION
                - :OWL-CARDI-RESTRICTION
              - :OWL-MINCARDI-RESTRICTION
                - :OWL-CARDI-RESTRICTION
              - :OWL-SOME-RESTRICTION
            - :OWL-LOGICAL-CLASS
              - :OWL-COMPLEMENT-CLASS
              - :OWL-INTERSECTION-CLASS
              - :OWL-UNION-CLASS
          - owl:Class
      - :SLOT
        - :STANDARD-SLOT
          - rdf:Property
            - owl:DatatypeProperty
            - owl:ObjectProperty
      - :FACET
        - :STANDARD-FACET
    - :CONSTRAINT
      - :PAL-CONSTRAINT
    - :ANNOTATION
      - :INSTANCE-ANNOTATION
    - :RELATION
      - :DIRECTED-BINARY-RELATION
    - :OWL-ANONYMOUS-ROOT
    - :OWL-ALL-DIFFERENT
    - :OWL-ONTOLOGY
  - :DIRECTED-BINARY-RELATION
  - Board
    - Junior_Fellow
      - Junior_Proctor

        - Provost
            - Vice_Provost
        - Senior_Fellow
            - Bursar
            - Registrar
            - Senior_Dean
            - Senior_Lecturer
            - Senior_Proctor
        - Statutory_Officers
            - Bursar
            - Junior_Dean
            - Junior_Proctor
            - Registrar
            - Registrar_of_Chambers
            - Senior_Dean
            - Senior_Lecturer
            - Senior_Proctor
            - Senior_Tutor
    - Body_Corporate
        - Additional_Member
        - Fellow
            - Junior_Fellow
                - Junior_Proctor
            - Senior_Fellow
                - Bursar
                - Registrar
                - Senior_Dean
                - Senior_Lecturer
                - Senior_Proctor
        - Foundation
        - Provost
            - Vice_Provost
    - Divinity_School_Council
    - Person
        - Additional_Member
        - Doctor
        - Master
            - Senior_Master_Non_Regent
        - Staff
            - Academic_Staff
                - Associate_Professors
                - Fellow
                    - Junior_Fellow
                        - Junior_Proctor
                    - Senior_Fellow
                        - Bursar
                        - Registrar
                        - Senior_Dean
                        - Senior_Lecturer
                        - Senior_Proctor
                - Junior_Lecturers
                - Other_Academic_Officers
                - Professors
                    - Class_A
                    - Class_B

- Class_C
- Class_D
- Class_E
- Class_F
- Readers
- Provost
  - Vice_Provost
- Statutory_Officers
  - Bursar
  - Junior_Dean
  - Junior_Proctor
  - Registrar
  - Registrar_of_Chambers
  - Senior_Dean
  - Senior_Lecturer
  - Senior_Proctor
  - Senior_Tutor
- Students
  - Scholars
    - Foundation
    - Non_Foundation
- Visitors
  - Chancellor
    - Pro_Chancellor
  - Nominee

o Senate
- Caput_Of_Senate
  - Chancellor
    - Pro_Chancellor
  - Provost
    - Vice_Provost
  - Senior_Master_Non_Regent
- Chancellor
  - Pro_Chancellor
- Doctor
- Master
  - Senior_Master_Non_Regent

o University_Council
o owl:Class
o rdf:Property
- owl:DatatypeProperty
- owl:ObjectProperty

Fig. 11: The class hierarchy produced using Protégé

Analysis

The class hierarchy shown above is the result of entering the relevant data in Protégé, converting it to OWL, and using the Generate HTML button in Protégé.
This hierarchy, and the OWL classes they describe, can be viewed at:

http://www.matrix.netsoc.tcd.ie/~cmandal/

The classes have been arranged from the most general, class *Person*, or class *Board*, to the most specific, class *Senior_Master_Non_Regent.*
As described above, each of these classes has slots, which describe the properties of any given class.

Abstract Classes

The classes *Board, Body_Corporate*, *Person, Divinity_School_Council, Senate,* and *University_Council,* are defined as **abstract** classes. That is to say that there can be no instance of these classes, but they exist as a combination of instances from other classes. For example, there is no instance of class *Board*, because the Board is not an entity upon itself, but rather a combination of the Provost, Fellows, and all the other people that make up the membership of the Board.

A Note on Class Cycles

Class cycles should be avoided in any class hierarchy. There is a cycle in a hierarchy if some class A has a subclass B and at the same time B is a superclass of A. This is tantamount to declaring that classes A and B are the equivalent.
In particular the statutes in question threw up many possible class cycles when one position would appoint someone in another position and then be subject to the decision of this newly appointed person. Great effort was made however to ensure class cycles did not occur in the resulting ontology.

One Class that was particularly difficult to define was the class ViceProvost. The reason for this was that the function of the Vice-Provost position was subject to change if the Provost was to become incapacitated for some reason. For a while they were two separate classes on the same tree level in the hierarchy, but eventually it was decided to have the Vice-Provost as a sub class of Provost, thereby suggesting the subordinate role of Vice-Provost but at the same time allowing it to fill the role of class Provost if necessary.

A Note on Capitalisation and other Notation Conventions

The readability of an ontology can be greatly improved if we use a consistent scheme of capitalisation for concept names. It is suggested in the documentation of Protégé 2000 to use a capital first letter for a class name and all small letters for a slot name, thus distinguishing *Faculty* and

faculty, for example. In the ontology described in his project this convention is stuck to, even though there is no slot with the same name as a class.

When a concept name contains more than one word (such as *Junior Fellow*) there are a few choices as to how to enter them.

- Use a space: Protégé 2000 allows for spacing in concept names – *Junior Fellow*.
- Run the words together and capitalise each new word – *JuniorFellow*
- Use an underscore or a dash – *Junior_Fellow*, *Junior-Fellow*.

In the ontology presented here, any concept name containing more than one word uses an underscore to connect the words.

A Note on Constraints

In this project, some limitations were decided upon so as to give as correct and informative a description as possible, without sacrificing the generality of the ontology, as such there were some additions that could have been made weren't as they would have been beyond the scope of this project.

One of the main tools of Protégé that was not implemented in the final results was the constraint mechanism, whereby only certain values can be entered in certain fields. The name of the faculty could be limited to the faculty names set out previously, for example. Equally the number of instances a class could have could be controlled, thereby adhering to conditions laid out in the statutes, like there can be no more than 6 pro-Chancellors, or that there must always be 70 scholars on the foundation at any time.

The reason that this was omitted from the final results was two-fold.

Firstly, an ontology such as this, while relating specifically to the statutes of Trinity College Dublin, could be moulded with minimum effort to fit the structures of countless other institutions around the world, with the adjustment of certain terminologies and class subordination. The beauty of the Protégé application is that is makes adjustments such as these relatively hassle-free.

However, if one was to include all of the constraints alluded to in the statutes, the ontology would become so specific so as to not be of any use to anyone else. It would probably be as efficient to just build a whole new ontology.

This point of time-management is relevant to the second reason for omitting these constraints. Quite simply, to put into effect an exhaustive and complete set of constraints for this ontology, would be such an undertaking of such magnitude that it would take a team of domain experts and developers many months to agree upon these constraints and what form they should take, and finalise them. This certainly possible and would yield an extremely detailed result, and when coupled with a definitive set of instances based on the current incumbents of all of these positions in college would create a database infinitely more effective than anything on today's web, but was deemed beyond the scope of a final year project.

Queries

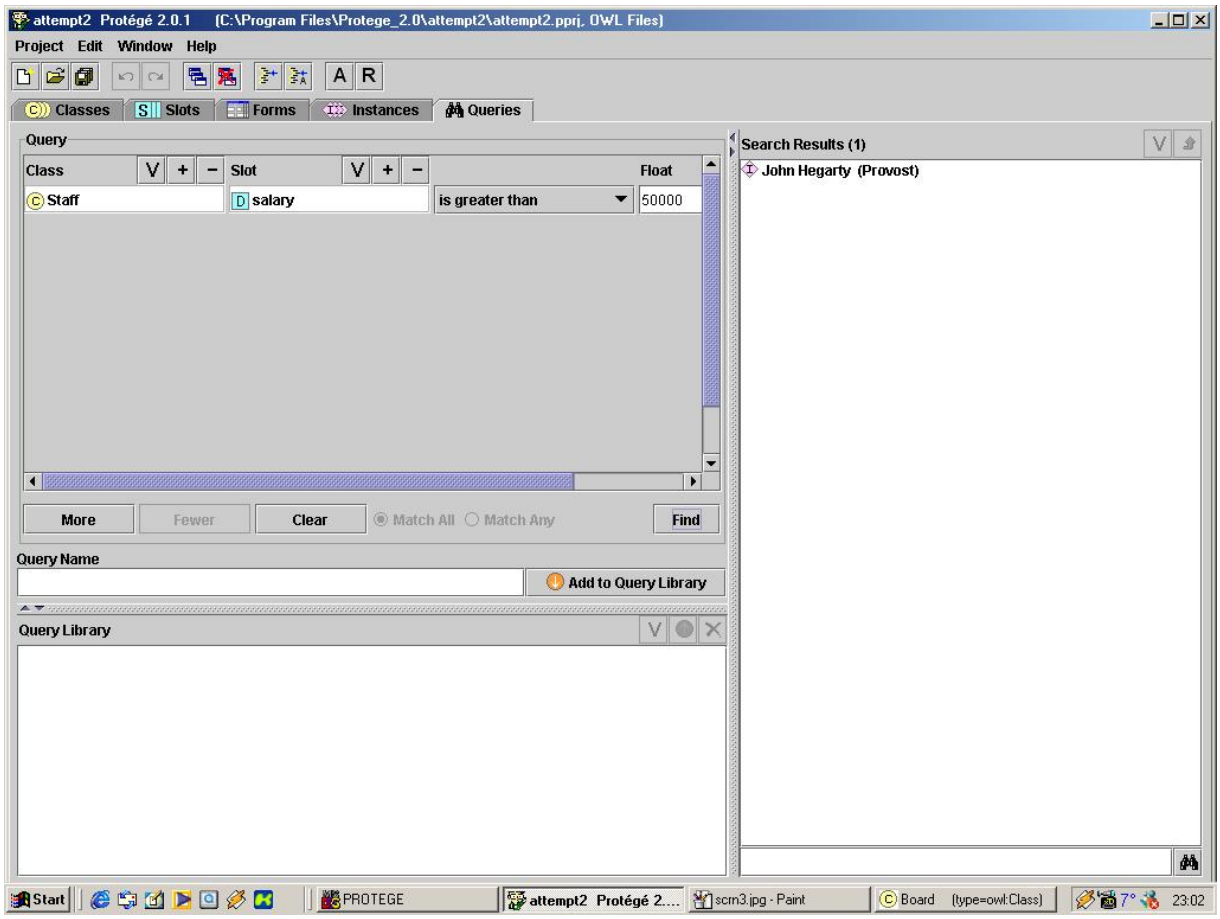
As explained before, there are certain **competence questions** that should be considered when building an ontology. Questions like 'Who will be turning 65 in the next year?" or "How many scholar positions will become available in 2007?" should yield accurate answers when carried out on this ontology.

In a normal HTML file containing the details of all of the people currently holding positions in the college including their salary and age it would still not be possible do a semantically dependent search. For example, a search for all of the members of staff earning more than €50k would not be possible.

However, using the protégé querying tool, it is possible to do just that. Below is a screenshot of this very search, with the result on the right

telling us that Provost John Hegarty is the only staff member earning more than this sum.

n.b. - The result of this search, as well as the data used for any of the instances is fictional.



(Fig. 12: A protégé query for members of staff with salaries above 50k)

A query as specific as this would not be possible on the syntactic web as it exists today.

The possibilities suggested by this ontology are far-reaching. If one was to input accurate instances for every class, the current statutory officers, scholars, fellows, masters, doctors, professors and so on, the result would be a source containing information on all of the salaried members of the

University that would be subject to a wide range of queries, and could be used as a source for simple enquiries to satisfy curiosity, or more practically as a tool for administrative staff to keep track of who is in what position and doing what. One could find out who needs to be replaced next year (due to turning 65 – the retirement age, for example) at the touch of a button.

About this Project

As regards the research done throughout the course of this project, it would be remiss to not mention some factors that influenced the direction that this project took.

At the outset, the ultimate aim of this project was to look into the possibility of devising a student enquiries website so with an emphasis on the disciplinary procedures laid out in the statutes providing for an occasion when a breach of student discipline is alleged.

There were multiple factors that swayed the eventual result towards what we have ended up with - an ontology for the power-structure of the University.

Firstly, to design an OWL ontology of any sort assumes a certain amount of previous knowledge. A grounding in HTML and XML, by which I mean ability and experience of both understanding and authoring source code, is pretty much taken as read in the majority of explanations of OWL available. This is understandable and logical, as you must learn to crawl before you can walk. However this writer had no previous experience in either HTML or XML, not to mention ontology-building or using Protégé 2000.

An accelerated course in HTML was used to familiarise some ideas such as the use of tags and so forth. This was extended to XML using various resources on the internet which were not documented at the time, and are thus not included in the bibliography section of this project.

As noted previously, these are the two types of people required to build an ontology – developer and domain expert. Thanks to the research conducted for this project this writer can now fulfil both of these roles for the statutes of Trinity College Dublin, and the role of developer for any given ontology.

Conclusions

Based on the results of this research, the capacity for precise querying resulting from building an ontology to represent the various positions provided for in the statutes, it is clear that the potential of the world wide web has still yet to be fulfilled. Over the next few years, countless documents on the web will be rewritten to provide semantics so that relevant material can be found more easily than is currently the case.

The statutes ontology that we are left with is part taxonomy part topic map. It can be identified as a taxonomy from its clear hierarchical structure, albeit imbued with a certain amount of semantic richness, through the various slots in each class and constraints assigned to certain classes.

It could also be called a topic map because each class, although organised in the class hierarchy, refers to a particular section of the statutes. In this sense each class can be described as a topic. The main shortfall of this ontology in terms of it being a topic map is that topic maps in general are created to merge indexes of information. Since theses statutes were looked at in isolation there was no way to try to merge it with another legal document.

The development of this ontology led me to some negative conclusions. The real difficulty in creating this ontology was that all of the available documentation – books, websites, presentations and so on – focussed on hierarchies of actual physical entities, such as the Elephant example, or

the Newspaper example (Editor → Journalist etc.) or the Employee example (Person → Employee → Manager). These were readily identifiable hierarchies, consisting of actual physical things being compared and categorised.

In the statutes however, as in all legal documents, the important information is not facts about physical things, but ideas and notions, procedures. How does one create a class for a procedure or for an abstract notion? There is an abstract class provided for in Protégé, however this writer could not see how to utilise this to overcome the real problem in this statutes ontology, ie how to define classes based on non-hierarchical, totally abstract ideas; concepts that were independent of each other and not readily isolated and formalised.

I was also surprised that the more I read on the subject, the more I realised that, at the moment, most of the books contain very similar information. The examples in each book were almost identical, and a lot of presentations made about the semantic web and in particular about the Protégé ontology builder use screenshots from the Protégé manual, not ontologies that have been built by the speaker!
This is clearly a consequence of the written word not keeping up with the developing technology, but can be unsettling to a researcher all the same.

In the semantic web, some benefits are clear, such as those explained in Berners-Lee's vision. However, in trying to create an ontology for the college statutes, I frequently found myself asking: "Why am I doing this? What is the benefit of this ontology?"
The query tool, as explained already, is an obvious pro, with the ability to search instances for certain slot values quickly. Links to relevant points in

the statutes are of course useful too, so that we can see in class Provost, for example, a link to the relevant passage in the statutes and go there without having to trawl through the text.

However in a legal document, I remain unconvinced that a machine-readable semantic model can ever be any substitute for a human reading and understanding of the ideas and notions - the “spirit” of the law as opposed to the letter - behind the wording of statutes, or a constitution, or any similar document.

Another stumbling block that currently hinders and will continue to hinder the expansion of the semantic web is the lack of authoring tools allowing the ‘part-time’ web author to publish semantically described material. This writer, with 3 years of computer science study behind him, had to undergo considerable training and education about the concepts underlying ontology-building, the philosophy of Ontology, the limitations of the syntactic web, and all this before even attempting to apply this knowledge to the statutes.

For HTML there are a multitude of ‘translators’ available that will convert a formatted word document, for example, to HTML. OWL is by its very nature so much more complex and precise that a mainstream translating tool like those available for HTML is difficult to imagine.

Until a way is found to make semantic web authoring accessible to the hobbyists and ‘part-time’ web authors, it will remain the preserve of a small number of OWL-literate scientists, and this in turn will stop it from becoming an influence on people’s daily lives as envisaged in Tim Berners-Lee’s seminal paper.

Bibliography

Daconta M., Obrst L., Smith K.: The Semantic Web (2003)

Fensel D. et al: Spinning the Semantic Web (2003)

Gangemi, A., Mika, P.: Understanding the Semantic Web through Descriptions and Situations. Vrije University Amsterdam (2003)

Gruber, T.: Toward Principles for the design of Ontologies Used for Knowledge Sharing. Kluwer Academic (1993)

Horrocks, I., Sattler, U.: Logical Foundations for the Semantic Web

Lewis, D.: Introduction to Ontology-based Semantics

Moore, M.S.: Legal Reality: A Naturalist Approach to Legal Ontology. Law and Philosophy 21 (2002)

Noy, N., McGuinness, D.: Ontology Development 101: A Guide to Creating Your First Ontology. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880 (2001)

Ogden, C., Richards, I.: The Meaning of Meaning - A Study in The Influence of Language upon Thought and of The Science of Symbolism (1923)

“protégé.stanford.edu” – Protégé 2000 homepage

Rosch, E.: Principles of Categorization. Rosch and Lloyd (1978)

Walsh N., A Technical Introduction to XML. www.xml.com (1998)

“www.daml.org” – DAML homepage

“www.w3c.org” – W3C homepage

“www.xml.com” – XML learner’s webpage

The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation. The first steps in weaving the Semantic Web into the structure of the existing Web are already under way. In the near future, these developments will usher in significant new functionality as machines become much better able to process and "understand" the data that they merely display at present. The essential property of the World Wide Web is its universality.Â Agents should be skeptical of assertions that they read on the Semantic Web until they have checked the sources of information. (We wish more people would learn to do this on the Web as it is!)