Implementing a Knowledge Database for Scientific Control Systems

Daniel Gresh
Wheatland-Chili High School
LLE Advisor: Richard Kidder
Summer 2006

Abstract

A knowledge database for scientific control systems is a powerful tool that allows users to obtain comprehensive information from a single source. A Semantic Web implementation of a knowledge database is intelligent enough to recognize what the user wants to know and present the correct information back to the user. A solid foundation for such a Semantic Web database has been built using the Java programming language, as well as the XML, RDF, OWL, and SPARQL languages. Using this database, a user may already obtain specific, detailed information regarding any aspect of the OMEGA EP Control Systems in an efficient manner from a single source, as opposed to searching through numerous technical and design documents. This foundation of the Semantic Web implementation covers Oracle/SQL, as well as the XML/RDF/OWL/SPARQL aspect.

The potential for such a database implementation is substantial. Upon completion, the database will automatically gather information concerning scientific control systems, organize it in an intelligent manner, and present intelligent results back to a user when he or she asks the database a question. Further research into the concept of this Semantic Web implementation will lead to automation of data-gathering; a basic form of artificial intelligence; a single repository for important subject data; and the ability for users to input their own information to expand the database.

Introduction

The Semantic Web

The Semantic Web is a vision for the future of the Web and for the future of information exchange. Rather than simply having information presented to humans, information will be processed in a way that makes it easier and more efficient to find.

In the Semantic Web, metadata, or “data about data”, is very useful, as it allows machines to process information rather than present information. By processing information, a machine

is able to gather enough useful information to solve problems, which greatly assists any users of such a system.

Due to this ideal, a Semantic Web implementation of a knowledge database will allow a user to find exactly what he or she is looking for with minimal effort and high efficiency, as the information in the database will be processed so the database can present the user with the correct information.

Background

LLE has gathered an enormous amount of data and documentation since the start of operations. This data is currently spread over a number of different repositories. Currently there are several hundred active control subsystems, all of which gather and record subject data, which is documented in many ways and stored in one of the data repositories.

The current data repositories include the Product Data Management (PDM) system, word documents, spreadsheets, databases, images, schematics, and events in onboard control memory.

The current PDM system consists of a few general subtopics for the included documents, and then displays links to all the documents, which a user can click on to view a certain document. The PDM system also includes a simple Google search box in which a user can search for certain keywords, and a Google search will be performed on the numerous documents, returning the most relevant documents to the user.

The Oracle database is simply a store for all the data and image information related to OMEGA and OMEGA EP shot operations.

Spreadsheets and other documents are often isolated on personal office workstations.

Purpose of the research project

Easily retrieving data from one or any current LLE documentation requires expert knowledge of LLE infrastructure and of the subsystem. For a user to retrieve data using the current system, he or she must follow a number of steps.

For example: the user must determine which data source to browse for the information. The user must already know if the data he or she is looking for is located in the PDM system, or if the data is located in the Oracle database. If a user does not know which one of these sources the data is located in, he or she will have to browse through both databases, which is very tedious.

Second, the user must know where to find the data in a certain data source. If using the Oracle database, the user will

need to know which table he or she wants to pull information from. If using the PDM system, the user will need to know which document the information is located in. Otherwise, more tedious searching is required.

Third, the user must parse through the document or table he or she selected, and retrieve the correct data. While this is not a big problem with the Oracle database, it can be very problematic when using the PDM system. Not only does the user need to wade through numerous technical and design documents, he or she must parse the documents manually and extract the correct information. This may lead to even more tedious searching, as a document may not have the specific information a user is looking for, and he or she will have to parse through another document.

Purpose of a knowledge database

A Semantic Web implementation of a knowledge database helps solve this problem by organizing data into a hierarchical structure in which data is organized conceptually and is related to other data through conceptual means. This is also known as an ontology.

After organizing the data in this fashion, the data is stored in a single repository, from where it can be extracted easily. Important data from all sources -- e-logs, technical journals, requirements documents, etc. -- is stored in one location in an organized manner, rather than having the data spread over multiple data sources in which the data is loosely organized.

Furthermore, the data is easily extractable. Rather than manually parsing through numerous documents or tables in a database, a user is able to easily navigate to whatever concept he or she is interested in and view all the related data pertaining to that concept. Also, the Semantic Web implementation enables the user to search the knowledge database using freeform text, or natural language.

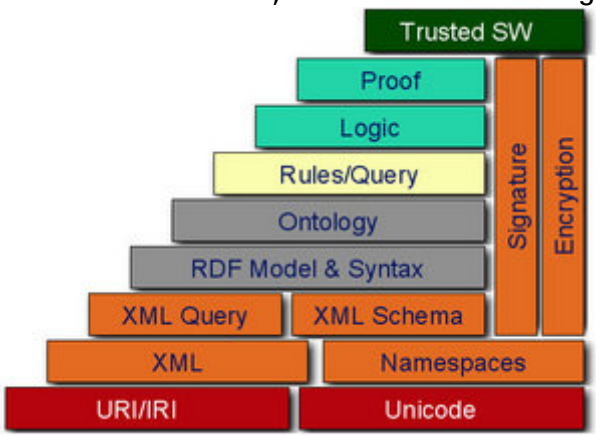

Figure1: The World Wide Web Consortium Semantic Web Framework (http://www.w3.org/2001/sw/)

As seen in Figure 1, the Semantic Web has many different building blocks. The basis of the Semantic Web is formed with URI/IRI, or resources. The resources are then structured in a certain way using XML and XML namespaces so the resources can be processed by machines. Then, XML Query and XML schema allow the data to be further refined and queried by providing a slightly more rigid structure; this rigid structure assists XML Query to find information. Finally, using RDF and OWL, which refine the data into an ontological structure, logic and proof are established, which allows machines to draw inferences from the resources, leading to a trusted, stable Semantic Web.

The knowledge database also lets users input their own information. The knowledge database receives manual input, and recognizes when an update to a document or e-log has been made, and reacts accordingly. For example, if a technician makes an entry into an e-log, or an engineer creates a new design document, the knowledge database recognizes this event, obtains the important information from the entry or document, and stores it in the database accordingly.

A Semantic Web implementation is a universal system for processing information. One of the many advantages a knowledge database of this structure has is the ability to connect to other knowledge databases and incorporate that knowledge into useful information. A database of this structure is able to not only contain important subject information from local sources, such as LLE control systems, but also gather and use data and information from external, related sources, further improving efficiency.

A knowledge database of this fashion has unlimited potential for expansion and upgrades. It is a universal repository for important information; this information can be easily extracted due to the data being stored in a machine-readable format.

Furthermore, the tools required to build a knowledge database of this fashion are very easy to acquire and easy to use. Also, the majority of the tools used for this work are open-source tools (free for anyone to use and develop). One of the tools used for this work was not open-source; however it is possible to create a knowledge database with an open-source tool that provides the same features.

The goal of this project was to build a strong, working foundation for a Semantic Web implementation of a knowledge database. This goal was accomplished; an RDF/OWL database was created using Java and the Jena development package for Java, and it is queried using SPARQL through Java and Jena.

Research and Analysis

Researching the implementation of a knowledge database that would store the data in a single repository required choosing a small subset of controls. A few diagnostic and control subsystems were chosen for the test case. Much of the implementation used the Calorimeter system.

Preliminary research and analysis

One of the goals of this project was to analyze methods for creating a knowledge database, and decide which method would work best for the project. Then, the appropriate techniques and tools would be used to create a strong foundation for the knowledge database. Furthermore, the research and analysis was to be done independently. Any tools and/or methods could be chosen that would create a knowledge database in the most efficient manner.

Initially, fourteen different languages were investigated. Eleven of them are listed in Table 1, together with their purpose, and the reason why they were or were not selected for use.

Many different tools and software packages were analyzed that might be useful, including Altova SemanticWorks, Protégé OWL, Jena, Lixto, Inxight, Evermap, Adobe Online Tools, KAON, JTP, Twinkle, Metadata Miner, Pellet, Omnipage, PDF Extraction, PDF to all, PDF Extractor API, pOWL, OWL API, PDF Metadata, VorteXML, and XML Army Knife. Tools were looked at for writing RDF/OWL, for performing SPARQL queries, for writing PHP, and for extracting data from documents.

Developing a data mining tool was a large concern. In order for quality, reliable data to be extracted from the numerous documents and sources that exist at LLE, some type of extraction tool would have to be developed. To determine how to create such a tool, the field of data mining and extraction was investigated. Studies of data mining and extraction methods led to one conclusion: the tool would have to be custom developed specifically for LLE; a pre-existing data mining tool would not work. This tool will be the subject of future work.

The World Wide Web Consortium (W3C) Web site proved to be a very valuable source of information,[1] in particular on the languages of RDF, OWL, XML, and SPARQL. Work related to this project is found in Refs. 9-12.

Language	Used?	Purpose	Reason
XML	Yes	Structure basic elements and resources	XML is the basis for RDF/OWL
XPATH	No	Navigate through XML documents	Navigating through XML with XPATH is not needed
XSL/XSLT	No	Transform XML into XHTML for readability	Presenting readable XML is not needed
XHTML	No	Design web pages with a stricter syntax than HTML	Database foundation was built in a Java GUI; no web page was needed
PHP	No	Server-side scripting language	Java has more resources for writing RDF/OWL
RDQL	No	Querying RDF data	Replaced by SPARQL
DAML/OIL	No	Designing ontology structures	Replaced by OWL
XQUERY	No	Querying XML documents	Not needed; SPARQL was used to query RDF/OWL
RDF	Yes	Describing resources	Provides basic structure for ontology
OWL	Yes	Relating resources to each other	Defines relationships between resources in ontology
SPARQL	Yes	Querying RDF/OWL	Queries RDF/OWL ontologies and extracts data

Table 1: Analysis of languages considered in this work

RDF/OWL and the Semantic Web

Using the languages of eXtensible Markup Language (XML), Resource Definition Framework (RDF), and Web Ontology Language (OWL), information and data in the Semantic Web will be organized in a structured manner in formats called ontologies. Ontologies have the ability to organize data and present it in a manner so that machines can draw conclusions and inferences from the data.

The Semantic Web is not pure artificial intelligence. Rather, as previously stated, it will use data that can be processed by machines to solve questions and perform tasks.

The three “main” languages that the Semantic Web concepts are based upon are XML, RDF, and OWL. These languages work together to describe resources and relate them to each other so that machines can process the information. These languages do not exist to provide information to humans, but rather to provide information to machines, so the machines can then determine what information is useful to provide to humans.

By structuring the resources using RDF/OWL and creating an ontology, numerous resources can be related to each other so they can easily be processed by a machine. An example of such a use is shown in Figure 2, which represents a small subset of OMEGA EP control systems. “IR Cal System” forms the base for the subtopics under it. These subtopics branch off into many different subtopics. However, due to the reasoning capabilities of RDF/OWL ontologies, although a subtopic such as “TSF Cal 1” may be a direct subtopic of “TSF Cals”, a machine can reason that “TSF Cal 1” is also a subtopic of “IR Cal System”.

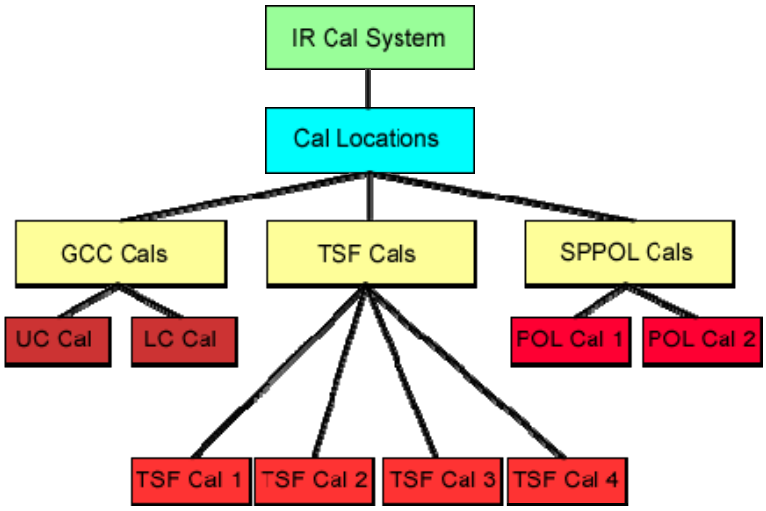

Figure 2: Structure of resources.[3] Cal: Calorimeter; GCC: Grating Compression Chamber; UC: Upper Compressor; LC: Lower Compressor; TSF: Transport Spatial Filter; SPPOL: Short-Pulse Polarizer; POL: Polarizer

RDF is structured as a series of graphs called “triple graphs”. These triple graphs, or triples, are what make it possible for machines to process RDF. A triple is a basic description that is read in the form: subject, predicate, object. For example, one could use RDF to specify the title and author of a certain book, by creating a triple with subject Example Book; predicate author; and object John Smith. This would then be read: the **author** of **Example Book** is **John Smith**.

Triples are instrumental in creating a database structure for LLE data. By using the triple format, data is stored in RDF where it can easily be extracted with SPARQL. For example, by creating a triple with subject “SPPOL Calorimeter Location”; predicate “information”; and object “The SPPOL Calorimeter is located between beamlines 1 and 2”, one could read: the **information** of **SPPOL Calorimeter Location** is, “**The SPPOL Calorimeter is located between beamlines 1 and 2**”.

Planning and Design

Preliminary design

Java was chosen as the language to program the database. The database had to be flexible, easily modifiable, and transportable across the Web to a variety of operating systems for easy access.

Java is an object-oriented programming language. In other words, an application is a series of individuals, or objects, rather than simple commands to the computer.[3] Using Java it is relatively simple to keep a database structure organized and functional.

Java provides a simple, object-oriented structure, whereas PHP does not. PHP is a server-side scripting language. There are tools available that allow RDF/OWL ontologies to be built using PHP, but Java’s object-oriented nature allows the RDF/OWL to be built in a more efficient, modifiable manner.

Furthermore, there are more resources available to assist in building RDF/OWL ontologies with Java than there are with PHP. RAP, the most common PHP development kit for writing RDF/OWL, has far less support and usage than Jena, the most common Java development kit for writing RDF/OWL. Because of this, using Java allows the programmer to learn the structure and functions faster, resulting in greater efficiency.

Tools

To develop an RDF/OWL database in Java, the Jena development package was used[9]. This is an external library for Java that allows for the programming and development of RDF/OWL databases. Jena also has support for the SPARQL querying language in its Application Programming Interface (API), which greatly assists when performing SPARQL queries against the RDF/OWL database.

Furthermore, some free tools were downloaded that would allow for the easier development of a Java application to create an RDF/OWL ontology. These include the Eclipse and Netbeans

Integrated Development Environment (IDE), as well as the Pellet OWL DL syntax checker/reasoner. A reasoner for OWL is a tool that determines how many inferences can be made from the ontology.

Current Design

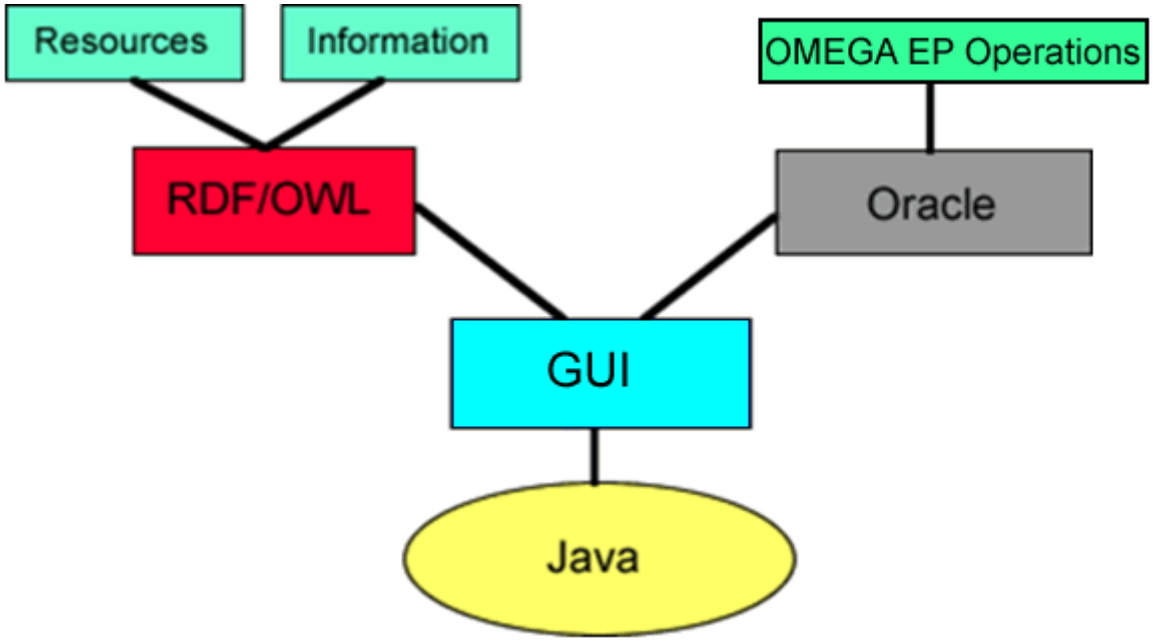

Figure 3: Database Design[3]

Figure 3 depicts the structure of the Semantic Web implementation. Java acts as the backbone for the knowledge database. From Java, a Graphical User Interface (GUI) is displayed, which contains access to the RDF/OWL and Oracle portions of the database. From there, resources and information pertaining to control systems, and resources and information related to OMEGA EP operations, are respectively displayed.

To create a solid structure, numerous Java classes were created. Java classes allow for simple portability and ease of coding. Having the entire application divided into a number of classes will also assist future programmers in learning the code and structure.

There are a few basic classes. One class creates the RDF/OWL ontology. Another class performs queries against the database, and another creates a GUI to display the information. In the future, many classes will be added, and the current classes will be refined into easier-to-use classes.

Current Status

Currently, a strong foundation for the database has been created, and a working, functional prototype of the database was successfully demonstrated. The database is currently a simple Java Archive (JAR) file for ease of use, and will be uploaded to the local intranet.

Figure 4 shows the structure of the graphical user interface for the Semantic Web Knowledge Database. A simple tree diagram is contained on the left side that allows a user to select the topic he or she desires. When the user clicks on a certain topic in the tree diagram, the information associated with that topic is displayed, as shown in Figure 4.

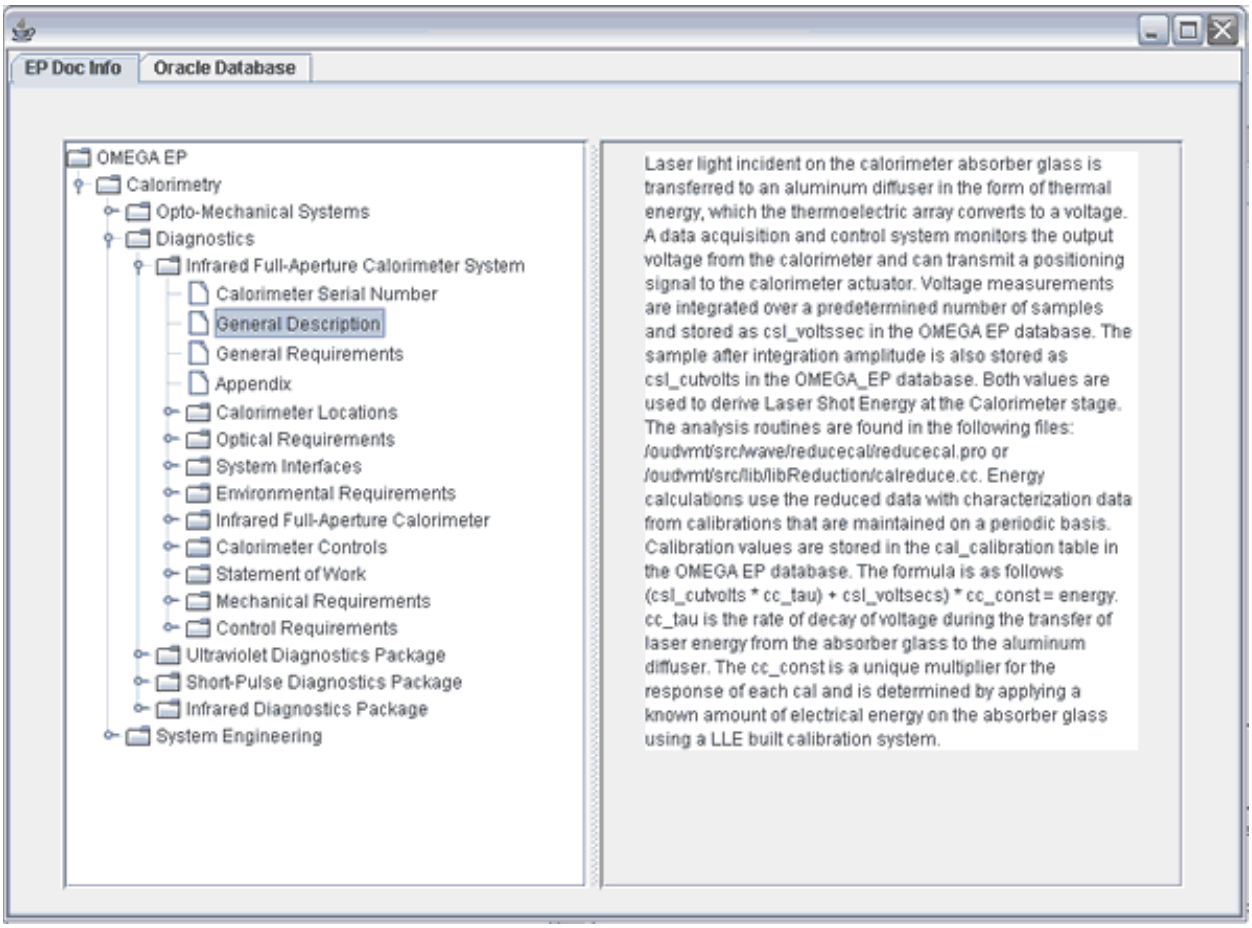

Figure 4: The Semantic Web knowledge database GUI

The Java code for the database has been created in such a way that it will be very easy to modify and expand upon. The code for the database has unlimited potential to evolve and expand, allowing multitudes of new features (see next section).

An ontology in OWL Full, for maximum ease of use, has been created that stores data about a small section of control systems: calorimetry. A GUI has been created that allows for easy searching of that database. Functionality has been added to the database that allows it to search the Oracle database about OMEGA EP operations.

The user currently browses through the information in the ontology by using a tree diagram. When the user selects a certain node on the tree, a SPARQL query is executed against the RDF/OWL ontology, which returns the necessary information and presents it to the user. The tree diagram allows the user to easily specify the information he or she is looking for, and view it in a simple, efficient manner.

Future Features

As the project moves forward, additional features will be designed and implemented. Further research into the field of artificial intelligence and the Semantic Web will lead to more advanced features that can be implemented at LLE. Much of this research may take place at MIT's Computer Science and Artificial Intelligence Laboratory, where extensive research is already being done.

In the coming year, a user will be able to enter freeform text questions into the database and receive the correct answer, rather than using a tree diagram. The database will gather all information automatically and will eliminate redundant information. Also, it will organize the gathered information in an ontology, where it can be structured in an intelligent manner. There will also be the ability for users to enter their own information into the database, expanding it with important information. The database will be able to recognize what information is important or not, and take action accordingly.

The database will be a single repository for important information. Rather than having numerous sources for data, all data will be stored in one central location where it can be easily and efficiently extracted. The database will have the ability to be expanded and upgraded to fit any reasonable need, and ultimately improve our knowledge of control systems and other important areas.

Acknowledgements

I'm excited to continue this fascinating project during my senior year at Wheatland-Chili High School. I thank LLE – in particular, Richard Kidder, Robert McCrory, and Stephen Craxton – for giving me this wonderful opportunity. I would also like to

thank Ivan Herman, the director of W3C’s Semantic Web Activity, and Jim Hendler, a professor at the University of Maryland, who is a chair on W3C’s Semantic Web Activity, for their assistance during the course of the summer.

References

1. http://www.w3.org/
2. http://en.wikipedia.org/wiki/Object-oriented_programming
3. PowerPoint presentation on the summer research project: http://www.seas.rochester.edu/~gresh/dangresh/DGreshRev3.ppt
4. http://www.w3.org/2001/sw/
5. NASA ScienceDesk white paper: http://ti.arc.nasa.gov/publications/pdf/ISWC-04%20SemanticOrganizer%20Paper%20pub.pdf
6. Additional notes and design ideas from notes taken over the summer: http://www.seas.rochester.edu/~gresh/dangresh/LLE_Semantic_Web.pdf
7. MIT’s CSAIL Decentralized Information Group Semantic Web Projects: http://dig.csail.mit.edu/
8. Jena: http://jena.sourceforge.net/
9. Personal Publication Reader: http://www.personal-reader.de/semwebchallenge/sw-challenge.html
10. NASA ScienceDesk: http://sciencedesk.arc.nasa.gov/
11. Platypus Wiki: http://platypuswiki.sourceforge.net/
12. Swoogle: http://challenge.semanticweb.org/