PUC

ISSN 0103-9741

Monografias em Ciência da Computação
n° 08/07

A Semantic Web Application Framework

Leonardo Magela Cunha

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO

RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22451-900

RIO DE JANEIRO - BRASIL

Monografias em Ciência da Computação, No. 08/07 ISSN: 0103-9741
Editor: Prof. Carlos José Pereira de Lucena June, 2007

A Semantic Web Application Framework

Leonardo Magela Cunha

leocunha@inf.puc-rio.br

Abstract. Documents have been the main vehicle of the Web until some years ago. With the advent of Web applications, data stored in organizations' databases or legacy systems has been made available to users. However, very often, the exchange of data between those applications themselves or between them and "end-users applications" were not possible since they used different formats for the information representation. The development of standards and the use of the eXtensible Markup Language (XML) solved parts of the problem. That was a syntactic solution and it works for several cases, e.g., schema interoperability in Business-to-Business e-commerce scenarios. Nevertheless, the lack of semantics on these data prevented applications to take more advantage of them. The idea behind the Semantic Web is to define explicitly the semantics of data available on the Web. Therefore, we expect another step forward where applications, being them corporative or for end-users, will "understand" the meaning of the data available on the Web. Once those applications can understand it, they will be able to help users to take advantage of this "data driven" Web and to perform their daily tasks easily. This report proposes a framework for the development of Semantic Web applications. Considering the scenario described in the previous paragraph, the number of possible applications that can be developed is almost infinite. For this reason, we restricted ourselves to examine the solutions that aim to solve the problem presented at the Semantic Web Challenge; and to propose a framework that represent those solutions. The challenge is concerned in demonstrating how Semantic Web techniques can provide valuable or attractive applications to end users. Our main concern was then to demonstrate and help a developer to achieve that value addition or attractiveness, through Semantic Web techniques, in a Software Engineering approach using frameworks.

Keywords: Semantic Web; Software Engineering; Semantic Web Applications; Frameworks; Semantic Web Challenge.

Resumo. Até alguns anos atrás, a Web disseminava, principalmente, documentos. Com o advento das aplicações Web, as organizações puderam disponibilizar informações que estavam em seus bancos de dados e sistemas legados. Entretanto, a comunicação entre estas aplicações ou com aplicações de usuários finais, às vezes, não era possível devido a diferenças no formato de representação dos dados. O desenvolvimento de padrões (standards) e o uso da eXtensible Markup Language (XML) resolveram muitos destes problemas. Apesar das soluções desenvolvidas serem somente sintáticas elas funcionam em muitos casos, como por exemplo, na interoperabilidade de esquemas em sistemas bussiness to bussiness de e-commerce. Entretanto, a falta do aspecto semântico impossibilitou que as aplicações fizessem mais uso dos dados ou os utilizassem de forma mais "inteligente". A idéia da Web Semântica é definir explicitamente o significado dos dados que se encontram na Web. Com isso, esperamos ter aplicações capazes de "entender" o que significam os dados da Web. E uma vez que estas aplicações entendam os dados, elas possibilitarão que os usuários finais utilizem essa nova Web "dirigida a dados" para facilitar as suas tarefas rotineiras. Esta monografia propõe um framework para o desenvolvimento de aplicações para a Web Semântica. Considerando o que dito anteriormente, o número de aplicações que podem ser construídas é quase infinito. Portanto, nós nos restringimos a observar as aplicações que tem por objetivo solucionar o problema apresentado pelo Semantic Web Challenge; e propor um framework que represente estas soluções. O Challenge tem como principal finalidade demonstrar como as aplicações podem atrair e beneficiar o usuário final através do uso das técnicas da Web Semântica. Conseqüentemente, nossa intenção é possibilitar que o desenvolvedor de aplicações possa atingir essa atração e benefícios, através do uso das técnicas de Web Semântica e de Engenharia de Software, utilizando um framework para o desenvolvimento das aplicações.

Palavras-chave: Web Semântica; Engenharia de Software; Aplicações para a Web Semântica; Frameworks; Semantic Web Challenge.

* This work has been sponsored by the Ministério de Ciência e Tecnologia da Presidência da República Federativa do Brasil

In charge for publications:

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC-Rio Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22451-900 Rio de Janeiro RJ Brasil
Tel. +55 21 3527-1516 Fax: +55 21 3527-1530
E-mail: bib-di@inf.puc-rio.br
Web site: http://bib-di.inf.puc-rio.br/techreports/

Table of Contents

Figures List

Tables List

1 Introduction

The dreams of software that could "understand" data (on the Web) has been tackled by several approaches by researchers of different areas or fields, such as databases, semi-structured data, knowledge management, logics, formal representation and Web systems. Those dreams are not new. Additionally, in the last years, more and more data is available on the Web [W3C, 2005a] and "clearly" related through the linking capacity [Rossi *et al.*, 1999]. Also, the Web is distributed, dynamic, massive and an open world them were already addressed by an organization (World Wide Web Consortium - W3C) in the effort[1] to lead the Web to its full potential, through the development of protocols and guidelines including the development of the Semantic Web.

The Semantic Web aims to solve problems like interoperability, improvement of searching techniques, reliability in data, among others, by making formally explicit the semantics of the data. Adding semantics to the data available will permit applications to reason about the data and provide more personalized services to users [Berners-Lee, 1998] [Berners-Lee *et al.*, 2001]. According to the W3C Semantic Web Activity Statement: "The goal of the Semantic Web initiative is as broad as that of the Web: to create a universal medium for the exchange of data. It is envisaged to smoothly interconnect personal information management, enterprise application integration, and the global sharing of commercial, scientific and cultural data" [W3C, 2005b]. However, if it is possible, how is it done?

In particular, in the case of the Semantic Web, Fensel *et alli* [Fensel *et al.*, 2002] identifies that the following elements are required (Figure 1):

- formal languages to express and represent ontologies, which are, roughly, the artifacts that formally explicit the semantics of the data;
- editors to build, merge and reuse ontologies;
- reasoning services to enable advanced querying and help map between different terminologies;
- annotation tools to link unstructured and semi-structured information sources with metadata;
- tools for information access and navigation that enable intelligent information access for human users; and
- translation and integration services between different ontologies that enable multistandard data interchange.

[1] About the World Wide Web Consortium (W3C) - http://www.w3.org/Consortium/ - accessed: 26/09/2006.

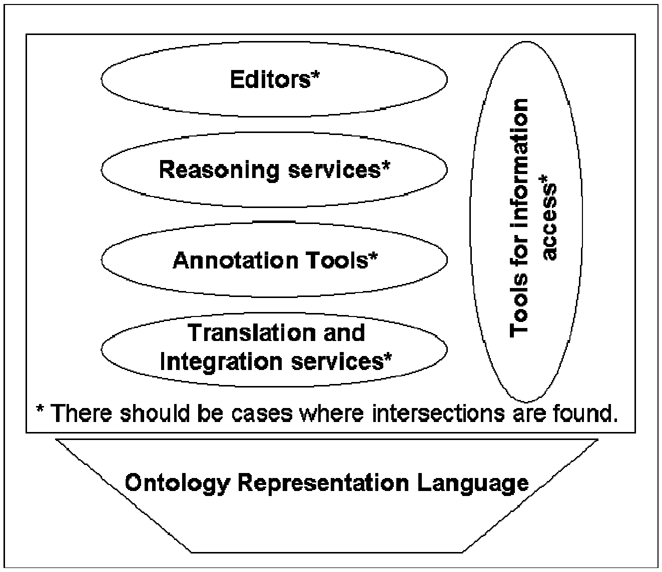

Figure 1 - An "interpretation" of Fensel *et alli* (Fensel, Hendler *et al*. 2002) elicitation of tools or technologies for the Semantic Web

Many of those elements (tools or technologies), in Figure 1, have been tackled by several researchers. However a question stills calls the attention, that is, how the "tools for information access and navigation that enable intelligent information access for human users" "look like", and more, how to develop them? As pointed by Alavi and Leider [Alavi & Leider, 1999], knowledge management systems (KMS), as the tools in the question, have to deal with different capabilities such as information-based, technology-based and culture-based ones. It is also clear that no dominant technology or tool (such as browsers, videoconferencing tools etc.) or product for KMS emerged in their survey that supplied all those capabilities.

1.1 The Problem

According to Conallen [Conallen, 1999], the differences between a Web site and a Web application involve its usage. In Web applications, the developers should focus the modeling effort on the business logic and business state without paying less attention to presentation details. However, something to be strived for is the separation of business and presentation concerns. If the presentation concern is important or complex, of course it should also be modeled but not as part of the business concern.

If we consider the characteristics of Web applications, the characteristics of the Web raised in [Heflin *et al.*, 2003] seems even more pertinent. For them the Web is:

- distributed: there is no centralized authority;
- dynamic: data can be, and often is, out of date;
- massive: an issue of scalability. We have to restrict expressivity or use incomplete reasoning algorithms;
- open world: information can be, and often is, incomplete.

Turning our focus to the Semantic Web again, designing and implementing a Semantic Web application (SWAPp) requires lots of pragmatic decisions [Tummarello & Morbidoni, 2005]. This work deals with the question of how the SWAPps "look like" and how to develop them. We then are interested in understanding and restraining the significance of which aspects are behind or supporting those applications. The answer to those questions can lead: end-users to better understand the benefits of the SWAPps; and, developers to take the pragmatic decisions in a conscientiously manner.

For the end-users, the benefits from using Semantic Web techniques or technologies should be transparent. On the other hand, for the developers, it is important to understand how those techniques or technologies relate to each other and which decisions have to be taken in order to achieve the benefits offered by the "new" explicit semantics of data.

If the pragmatic decisions taken by the SWAPp developers follow a Software Engineering approach, this approach will show the way to better software that is reusable, portable, maintainable, dependable and efficient. Therefore, in the next section, we outline our approach for answering the questions of how a SWAPp "looks like" and how to develop it.

1.2 Proposed Solution

To answer the question of how the Semantic Web applications "look like" and how to develop them, this work will review the applications submitted to the Semantic Web Challenge[2] (SWC). The SWC is concerned in demonstrating how Semantic Web techniques can provide valuable or attractive applications to end-users. As we shall see in Chapter 3 , the challenge shares some of the same objectives as this work. The review of the applications will present some possible realistic alternatives to the pragmatic decisions that have to be taken by one that wants to develop a SWAPp.

By reviewing the applications, we restrain the domain of this work to the same domain of the challenge. Therefore, our approach is limited by the SWC domain, that is, it is not applicable to all the Semantic Web applications that a developer could implement. However, the range of applications as defined by the challenge is already broad enough. That is true because the organizers of the challenge define some broad minimal and desirable requirements to characterize a SWAPp.

With the review of the applications, we propose a domain analysis of the submissions to the SWC. Based on this domain analysis of the applications, we define a set of types of application and functionalities offered by them. This will serve as one of the requirements for developing a framework for SWAPps. In the next section, we decompose the proposed solution into objectives so that they become more feasible.

1.3 Objectives

The proposed solution to the question of how the Semantic Web applications "look like" and how to develop them, considering the domain of the SWC, lead us to the following objectives:

- to review applications submitted to SWC;
- to use a standardized way to register the review process;

[2] The Semantic Web Challenge - http://challenge.semanticweb.org/ - accessed: 16/06/2006.

- to perform a domain analysis of the applications based on the review process;
- to propose a framework based on the types of application and their functionalities discovered during the domain analysis; and
- to illustrate how the architecture of framework might be instantiated.

Up until the time of writing, 35 applications were submitted during the three first editions of the SWC. As stated before, the applications do not represent all possible applications on the Semantic Web. On the other hand, they do represent a segment of applications that satisfy specific requirements proposed by the challenge's organizers.

We will register the review process using an extended schema for describing projects (see section 3.3 for detailed information on the schema choice). Based on the information captured during the review process, we propose a domain analysis of the applications. This domain analysis will serve as one of the boundaries for the proposition of a framework for SWAPps.

With the proposition of the framework, we intend to provide assistance (or guidance) for the developers of a set of SWAPps, which is defined as a "valid" combination of functionalities offered by a type of application. The illustration of how the framework could be instantiated, through its architecture, shall illustrate the adequacy and relevancy of the framework.

1.4 Contributions

Based on the objectives defined, the contributions of this work are:

- the register, in a standardized form, of the review process used in this work of the applications submitted to SWC;
- the proposition of a set of types of SWAPps and their functionalities; and
- the presentation of a framework for SWAPps.

1.5 Related Work

If we consider the elements required to have the Semantic Web as defined in [Fensel *et al.*, 2003], our framework does not deal with the fundamentals of the Semantic Web, that is, it is not a tool to support ontology edition or storage. Our framework is neither an infrastructure application that offers general functionalities and, probably, access to tools that offer support for the fundamentals of Semantic Web like Sesame [Broekstra *et al.*, 2002] or Jena [Carroll *et al.*, 2004].

Our framework is then in an intermediary level between the infrastructure applications and end-user SWAPps. There are several works on this same level; however, we could not become to know of any dealing with the domain we chose. For example, Semantic Hypermedia Design Method (SHDM) is used for dealing with the development of hypermedia applications using Semantic Web technologies [Lima, 2003].

The differential of our work is that it is concerned with a very specific, still broad-ranging, domain: the SWC domain. In addition, our approach relies on some benefits from the use of framework such as reusability, portability, maintainability and dependability. Our framework is not either an end-user Semantic Web application since it represents a set of them that could be instantiated by its customization.

It is also important to remember the empirical aspect of our work. We reviewed 35 applications submitted to the challenge and performed a domain analysis based on them. Our framework has then the characteristic of using a "bottom-up" approach that offers the users of the framework with a set of potential pragmatic decisions already taken as a choice to implement their own SWAPps.

1.6 Summary

In this chapter, we have contextualized the problem of how Semantic Web applications "look like" and how to develop them. We have also shown how relevant those questions are and briefly introduced a proposed a solution to them. We decomposed this proposed solution into objectives that led to the contributions of this work. The main contribution is the proposition of a framework for SWAPps. We also presented some related works and how our approach differentiates from them.

The rest of this work is structured as follow: in the next chapter, we present some fundamentals about the Semantic Web. In the following chapter, we present the Semantic Web Challenge (SWC) and how we extended an RDF vocabulary to review the applications submitted to the challenge. The original vocabulary is presented in Annex A - The DOAP Vocabulary and the extended version of it is presented in Appendix 1 - The SWDOAP Vocabulary.

Chapter 4 presents the domain analysis of the applications submitted to SWC. This domain analysis is composed of definitions and examples of types of application and their functionalities. Chapters 5 , 6 and 7 present, respectively, the applications submitted to SWC in 2003, 2004 and 2005. In these chapters, for each application we also show the type of application and the functionalities it offers based on definitions presented in Chapter 4 .

Chapter 8 presents and discusses the proposed Semantic Web application framework (SWAPpFW). Chapter 9 presents the conclusions of this work as well as its contributions and related works.

2 Semantic Web

According to Berners-Lee [Berners-Lee, 1998] [Berners-Lee *et al.*, 2001], a definition to the Semantic Web is: "an extension of the Web obtained via the semantic addition to the present data format representation". The main purpose of having a Semantic Web is making the Web data understandable for humans and for software entities such as agents [Silva *et al.*, 2003] or components [Szyperski, 1998]. In this sense, if the Web content would be machine processable, applications could have access to a huge variety of resources, which could be shared, integrated and processed to produce a result with more value to the user.

The "basis" of the present Web is the HyperText Markup Language (HTML), which allows human-to-human communication, because humans can understand its pages content. Benjamins *et alli* [Benjamins *et al.*, 2002], present the Semantic Web as a mean of treating the problem of information overload caused by the continuous Web growth, in size, languages, and formats. In the Semantic Web, pages present not even a set of words, figures, tables and other elements, but the code and the structure of their meanings, allowing the electronic processing of it.

Formal representation of meaning can take a variety of forms. One of the oldest formalisms is semantic networks. A semantic network represents knowledge as a set of nodes connected by labeled links. The meaning is implied by the way a concept is connected to other concepts. Another approach are frames systems that are isomorphic to semantic networks [Heflin *et al.*, 2003]. A further way to facilitate the expression and justification of arguments would be through formal logics. In the many branches of logic, systems consist of:

- a well defined language for the representation of knowledge; and
- well defined methods for reasoning.

Those systems are limited in the type of knowledge that they can represent and in the type of reasoning that can be performed [Frost, 1986]. Hence, logicians developed other kinds of logics to avoid those restrictions. Examples of such branches of logic are predicate logic, first order predicate logic, non-monotonic logic and description logic among others. In the case of the Web, computational restrictions are one of the most important restrictions. That is one reason for the need to choose a specific knowledge representation formalism, e.g., a branch of logic, to implement the Semantic Web.

Once that formalism is chosen, some artifact will be defined to contain the code and structure of the meaning of the elements on the Semantic Web. That is, roughly, the role of an ontology. In the next section, we go further on the definitions of ontologies. The following sections present the relationship between the W3C and the Semantic Web and the controversialism about one of the architectural basis of the Semantic Web, the Semantic Web stack.

2.1 Ontologies

One of the most referenced definitions of ontology, in Computer Science, is due to Gruber [Gruber, 1993]. To him, an ontology is an explicit specification of a conceptualization. In this definition, by conceptualization we can understand the concepts, objects and other entities that exist in an area of interest, and the relationships between them. Borst [Borst, 1997] made a slight modification in Gruber's definition, and it seems more appropriated: ontologies are defined as formal specifications of shared conceptualizations.

Following Borst's definition, we can infer that ontologies are important to software systems that aim to search, combine or integrate information from different communities. This is exactly the case of Web information, where ontologies can allow the semantic representation of data.

This section intent is to provide ontology definitions. However, there is not a common definition for ontology in Computer Science. One of the reasons is the large spectrum of possible uses for ontologies [Breitman & Casanova, 2006]. That spectrum is depicted in Figure 2. For more details about each of the uses of ontologies, please refer to [McGuinness, 2003] or [Breitman & Casanova, 2006].

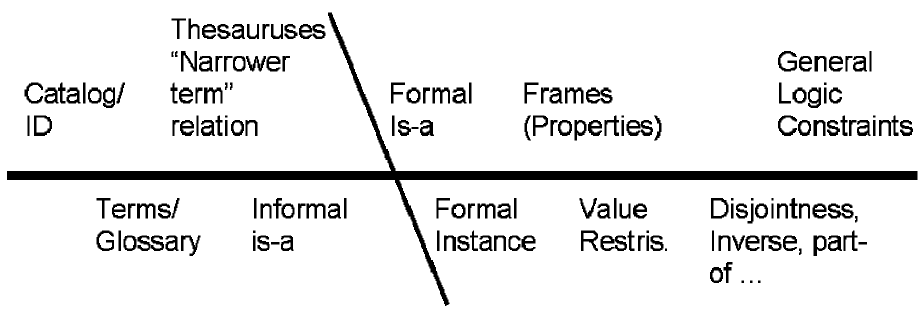

Figure 2 - An ontology spectrum [McGuinness, 2003]

Another way to understand the diverse uses of ontologies is through the understanding of what the term "semantics" means on the Semantic Web. Uschold [Uschold, 2003] provides an approach for that through a semantic continuum shown in Figure 3.

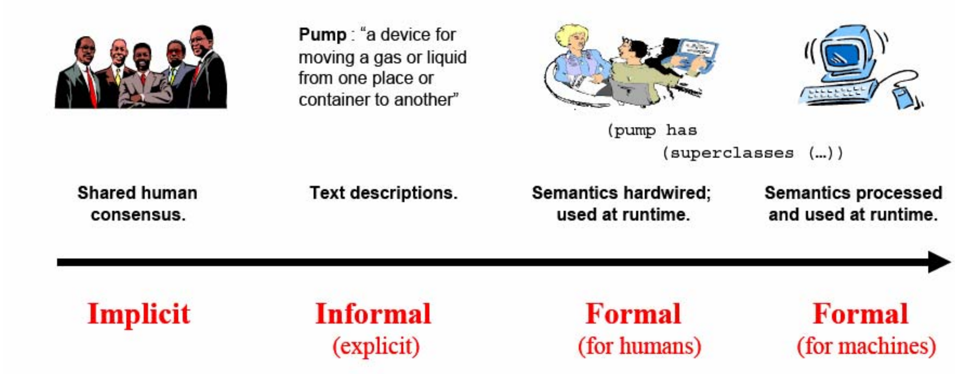

Figure 3 - Semantic Continuum …[3] [Uschold, 2003].

[3] Continuation of the caption on [Uschold, 2003] figure: "Semantics may be implicit, existing only in the minds of the humans who communicate and build Web applications. They may also be explicit

From the discussion above, it is clear that many ontology definitions may exist and they can be somewhat altered to accommodate a project or research area. We do not go further in that discussion because it is not the focus of this work. More information about definitions of ontologies can be found at [Guarino, 1998] [Guarino, 1997] [van Heijst *et al.*, 1997] [Guarino, 1995] [Guarino & Giaretta, 1995].

Ontologies, in the Semantic Web, are represented by the use of Web ontology description languages. Examples of Web ontology description languages that were developed are: Simple HTML Ontology Extensions (SHOE) [Heflin & Hendler, 2000], Resource Description Framework (RDF) [W3C, 2004e], RDF Vocabulary Description Language 1.0 (RDF Schema) [W3C, 2004f], DAML+OIL Language (DAML+OIL) [W3C, 2001], OWL [W3C, 2004b] among others. As these languages are based on the eXtensible Markup Language (XML), they are richer than HTML. The languages allow the representation of the structure of contents through their syntax and the representation of the semantics through ontologies to describe properties of or relationships between concepts. Some of the Web ontology description languages allow for inferences to be made about the concepts and relationships between these concepts expressed on the ontologies.

2.2 W3C and the Semantic Web

The World Wide Web Consortium (W3C) is an international consortium with the mission to lead the Web to its full potential by developing protocols and guidelines that ensure long-term growth for the Web.

According to the W3C [W3C, 2004g], some of the prior languages used to represent ontologies, elicited earlier in the previous section, and to develop tools for particular user communities were not compatible with the architecture of the Web in general, and, in specific, the Semantic Web. The consortium then proposed and recommended the Resource Description Framework (RDF) [W3C, 2004e] which is a language for representing information about resources in the Web. The RDF Vocabulary Description Language 1.0 (RDF Schema) [W3C, 2004f] was the next recommendation and step from W3C to represent ontologies.

Subsequently, W3C proposed and recommended the Web Ontology Language (OWL) [W3C, 2004b] that "extends" RDF and RDF Schema providing some capabilities to ontologies such as scalability; distribution; compatibility with Web standards for accessibility and internationalization; openness and extensibility. As stated before, only special branches of logic are computable. Therefore, OWL was designed to offer three increasingly expressive sublanguages [W3C, 2004a]:

- OWL Lite: supports, primarily, classification hierarchies and simple constraint features;

---

and informal, or they may be formal. The further we move along the continuum, the less ambiguity there is and the more likely it is to have robust correctly functioning Web applications. For implicit and informal semantics, there is no alternative to hardwiring the semantics into Web application software. In the case of formal semantics, hardwiring remains an option, in which case the formal semantics serve the important role in reducing ambiguity in specifying Web application behavior, compared to implicit or informal semantics. There is also the new possibility of using automated inference to process the semantics at runtime. This would allow for much more robust Web applications, in which agents automatically learn something about the meaning of terms at runtime."

- OWL DL: provides the maximum expressiveness without losing computational completeness[4] and decidability[5] of reasoning systems. OWL DL is named like that due to its correspondence with description logics [Baader *et al.*, 2003]. Description logics is a field of research that studies a particular decidable fragment of first order logic;
- OWL Full: offers maximum expressiveness and the syntactic freedom of RDF with no computational guarantees.

According to [W3C, 2004a]: "Ontology developers adopting OWL should consider which species best suits their needs. The choice between OWL Lite and OWL DL depends on the extent to which users require the more expressive restriction constructs provided by OWL DL. Reasoners for OWL Lite will have desirable computational properties. Reasoners for OWL DL, while dealing with a decidable sublanguage, will be subject to higher worst-case complexity. The choice between OWL DL and OWL Full mainly depends on the extent to which users require the meta-modeling facilities of RDF Schema (i.e. defining classes of classes). When using OWL Full as compared to OWL DL, reasoning support is less predictable". For more information about this issue see the OWL semantics document [W3C, 2004c].

Once the choice on which sub-language will be used in a solution is made, the question is what the advantages of such a choice are. In fact, the "use of ontologies by Web applications" or the "ontology understanding and processing by software agents" can be seen as a "way of building more intelligent applications in a near future while executing tasks in the closest conceptual level to the human level" [W3C, 2004d]. This last statement is very close to one of the objectives of the artificial intelligence area. However, as stressed by Breitman and Casanova [Breitman & Casanova, 2006], there is a distinction between artificial intelligence and the Semantic Web.

Artificial intelligence aims at constructing software that is capable of showing a level of intelligence that is similar (or superior) to human intelligence. On the other hand, one of the Semantic Web goals is to develop software that can help humans in making their decisions. Moreover, as stated by Uschold [Uschold, 2003], the implicit semantics, or shared human consensus, is "conceptually" far from the formal semantics processed and used at runtime by machines as depicted in Figure 3.

Besides these discussions, it is also desirable that applications become more secure and confident based on trusted ontologies and inferred information. The Semantic Web will enable even more interesting functionality through complex logics and the exchange of proofs to establish trust relationships [Hendler, 2001].

The recommendation of OWL and the previous assertions from what is expected from applications that use ontologies are illustrated in one of the architectural basis of the Semantic Web, which is the "Semantic Web stack" (see it in the context of Figure 4), first presented in a Berners-Lee's talk in XML 2000 Event [Berners-Lee, 2000]. For a definition of the layers, please refer to [Fensel *et al.*, 2002]. Nevertheless, the Semantic Web stack is controversial, and in the next section, we, briefly, report that controversialism.

---

4 All entailments are guaranteed to be computed.

5 All computations will finish in finite time.

2.3 The Controversialism about the Semantic Web Stack

Designing and implementing a Semantic Web application (SWAPp) requires lots of pragmatic decisions [Tummarello & Morbidoni, 2005]. Figure 4 depicts an example of that based on Berners-Lee's Semantic Web stack [Berners-Lee, 2000].

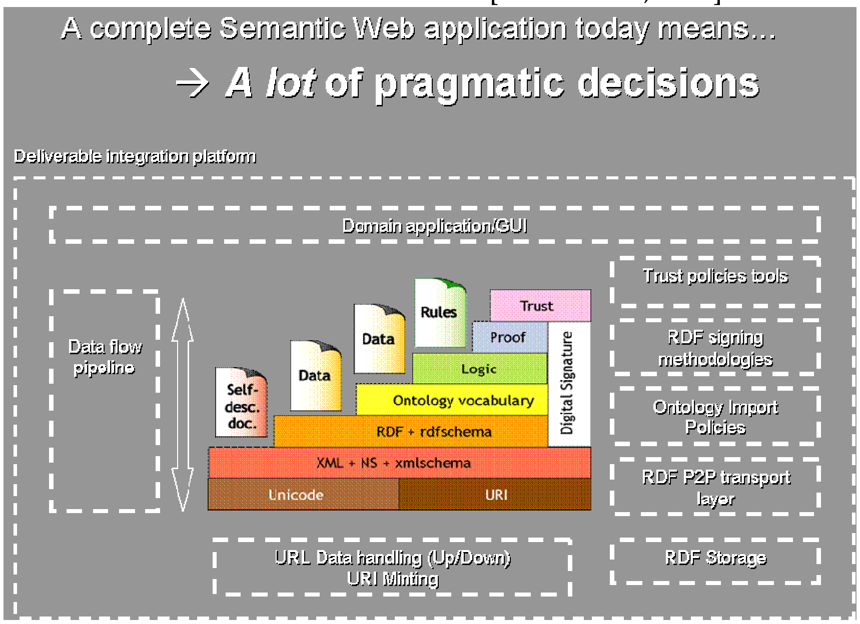

Figure 4 - Pragmatic decisions to design and implement a SWAPp based on Semantic Web stack [Tummarello & Morbidoni, 2005]

Nevertheless, the pragmatic decisions represented as dashed boxes in Figure 4 and an updated version of the Semantic Web stack proposed by Berners-Lee [Berners-Lee, 2005], presented in Figure 5, lead to some controversialism about the stack [Horrocks *et al.*, 2005] and [Patel-Schneider, 2005].

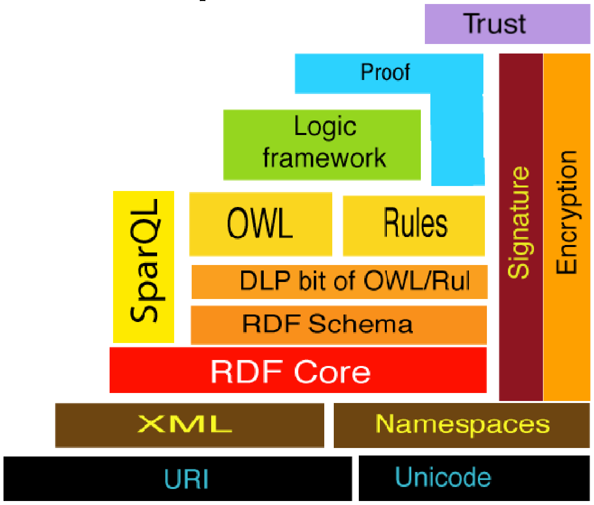

Figure 5 - An updated version of the Semantic Web stack

Both works, [Horrocks *et al.*, 2005] and [Patel-Schneider, 2005], present some misconceptions in the updated version of the Semantic stack. They also discuss and propose revised versions of the stack. The discussion and the proposed revisions are out of the scope of this work. However, as we are going to propose a framework for SWAPps, this controversialism has to be taken into account.

2.4 Summary

In this chapter, we presented the concepts about Semantic Web, ontologies, the relation between W3C and the Semantic Web through the recommendation of a Web ontology description language (OWL) and underlying languages. We briefly introduced, the controversialism about the Semantic Web stack, which can be considered one of the architectural basis of the Semantic Web. The majority of this chapter provides fundamentals to the contextualization of this work since we are going to propose a Semantic Web application framework.

We observed, in this chapter, that there is no universal definition for ontology and that some controversialism exists about the Semantic Web stack on its architectural layering of languages. Those two observations will be of importance when proposing the framework.

In the next chapter, we present the Semantic Web Challenge through its requirements and we extend an existing vocabulary to describe the applications submitted to the challenge.

3 The Semantic Web Challenge (SWC)

The Semantic Web Challenge[6] (SWC) had three editions (2003 [Klein & Visser, 2004], 2004 [Klein & Visser, 2005] and 2005 [Visser & Klein, 2005]) until the time of writing this work[7]. In the 2005 edition, the flyer of the challenge says that the general objective of the challenge is to apply "Semantic Web techniques" in order to build an "online application that integrates, combines, and deduces information needed to assist users in performing tasks". SWC was started at the International Semantic Web Conference[8] (ISWC) for answering questions like "What kinds of things can be realized with today's techniques? … Are any Semantic Web Applications out yet?"

The challenge does not purposely define specific data sets because the prospective applicability of the Semantic Web is very wide-ranging. However, concerns about distribution, portability and other characteristics of the Web are important here. In the SWC, there was not a previous definition of what an ontology should be, nor of the language that should be used to represent it. We could assume that this decision takes into account the same reason that no data sets are defined, that is, the applicability of the Semantic Web is very broad.

At least three members of the advisory board revise each application submitted to the SWC, named by the organizers as Semantic Web Application (SWAPp). The submitted applications have to attend the application definition presented in section 3.1 and the advisory board defines an additional goal each year for the challenge. We will present each year's additional goal in Chapters 5 , 6 and 7 where we describe and summarize each year's applications. In the rest of this chapter, we present the ranking of the applications and the discussion about how to describe the applications in this work.

3.1 Application requirements and desirable qualities

To define a SWAPp, a set of minimal requirements based on the discussion with several experts were elicited [Klein & Visser, 2004]:

- Considering the information sources of the applications, they must:
    - be geographically distributed;
    - have diverse ownerships - that is, there is no control of evolution;
    - be heterogeneous (syntactically, structurally, and semantically);
    - contain real-world data - that is, the sources must be more than toy examples.
- Considering the open/close world option: the application must assume an open world; that is, it assumes that the information is never complete;
- Considering the description of the data's meaning: the application must use some formal description.

---

[6] SWC - http://challenge.semanticweb.org/ - accessed: 16/06/2006.

[7] The 2006 edition of the challenge occurred in 2006, November 5th to 9th at ISWC06. This edition was not considered in this work since we did not have enough time to evaluate its submissions.

[8] ISWC - http://iswc.semanticweb.org - accessed: 19/06/2006.

Furthermore, additional desirable qualities were defined:

- Considering the data sources, they should:
  - be used for other purposes or in another way than originally intended;
  - exploit both static and dynamic knowledge - for example, a combination of static ontologies and dynamic workflows;
  - use the contents of multimedia documents.
- Considering users' access: multiple languages and access through devices other than a personal computer should be offered;
- Considering scalability: the applications should be scalable (in terms of the amount of data used and of distributed components working together).

3.2 Application Classification

There are several approaches, not considered in the challenge, for "somehow" classifying Ontology-Based Applications (OBAs), which are not necessarily designed for the Web, and SWAPps. For example:

- A Framework for Understanding and Classifying Ontology Applications [Jasper & Uschold, 1999] [Zyl & Corbett, 2000a] [Zyl & Corbett, 2000b];
- OWL Web Ontology Language Use Cases and Requirements [W3C, 2004d];
- Object Management Group[9] (OMG) Ontology Definition Metamodel (2nd revised submission) [DSTC *et al.*, 2005];
- OntoWeb's Successful Scenarios for Ontology-based Applications [Léger *et al.*, 2002].

On the other hand, the advisory board does not classify the SWC applications according to any categories, specifically, due to the broad-ranging objective of the challenge. However, the advisory board ranked the applications.

[9] OMG Homepage - http://www.omg.org/ - accessed: 21/08/2005

Table 1 - Semantic Web Challenge Summary

	2003	2004	2005
Number of Submitters	10	18	7
1st Prize	CS AKTive Space	Flink	CONFOTO
2nd Prize	SEmantic COllaboration (SECO)	MuseumFinland	FungalWeb
3rd Prize	Annotated Terrestrial Information (AnnoTerra)	SemanticOrganizer	Personal Publication Reader

Table 1 presents the number of submitters for each year and the name of the applications that won the challenge. Each application description can be found in the respective chapters of each year's challenge. In the next sections, we explain the foundations of our choice of how to describe the applications.

3.3 Describing the Applications

We reviewed and carried out a domain analysis of the 35 applications submitted to the Semantic Web Challenge in order to obtain elements to create a framework of SWAPps. Therefore, it is necessary to describe such submissions. In the next section, there is a short description of our main rationale on the choice of vocabulary to describe the submissions. Many of the applications submitters were invited to write an extended version of their abstracts submitted to the SWC. We tried to keep this work based on those papers, but sometimes it was necessary to consider other sources and papers as well.

3.3.1 The W3C's Applications and Demos Task Force at the Semantic Web Best Practices and Deployment Working Group

The Semantic Web Best Practices and Deployment[10] (SWBPD) is a working group within the Semantic Web Activity[11] in W3C. The aim of SWBPD is to provide developers of Semantic Web applications with practical support, ranging from engineering guidelines to educational materials. One of the working group's task forces is the Applications and Demos Task Force[12] (ADTF). It provides a documented list of SWAPps and demos to promote the Semantic Web and for use by developers.

On March 2005, ADTF members agreed upon a specific proposed criteria for applications and demos to be included in their list [W3C, 2005c]. The criteria for inclusion were:

- Only applications and demos with their own Description Of A Project (DOAP) metadata (see section 3.3.2) will be included;
- Only freely downloadable applications and demos will be included unless they are products of a W3C member;
- Only RDF, RDF Schema and OWL applications will be included.

In the face to face meeting minutes [W3C, 2005c] the explanations for the selection of such criteria are presented. Therefore, we are going to use DOAP descriptions for the SWC applications review. Since we are using DOAP descriptions, we present the DOAP project[13] in the next section. We will not follow the other criteria due to the SWC's characteristics. This decision requires the definition of an extension of the DOAP vocabulary, presented in section 3.3.3 . The SWBPD working group was closed[14] at 2006, September 29th. Its remaining activities were redirected to or evolved into other working groups.

---

[10] SWBPD - http://www.w3.org/2001/sw/BestPractices/ - accessed: 29/11/2005.

[11] W3C Semantic Web Activity - http://www.w3.org/2001/sw/ - accessed: 29/11/2005.

[12] ADTF - http://esw.w3.org/topic/SemanticWebBestPracticesTaskForceOnApplicationsAndDemos - accessed: 29/11/2005.

[13] The DOAP Project - http://usefulinc.com/doap - accessed: 16/06/2006.

[14] Semantic Web Best Practices and Deployment Working Group now closed - http://lists.w3.org/Archives/Public/public-swbp-wg/2006Sep/0014.html - accessed: 2006-11-10

3.3.2 DOAP: Description Of A Project

DOAP is a project to create a XML/RDF vocabulary to describe open source projects. The DOAP vocabulary is an RDF Schema similar to the Friend Of A Friend (FOAF) vocabulary [Brickley & Miller, 2005]. According to Dumbill [Dumbill, 2004a] [Dumbill, 2004b] [Dumbill, 2004c], the DOAP vocabulary is meant to be extensible and in his vision some semantics can be left behind in order to have a more “human-readable” schema. However, many “design decisions” expressed in [Dumbill, 2004a] would become formally defined using an ontology instead of a RDF Schema vocabulary. Moreover, that ontology would still have not a significant level of complexity or expressiveness.

The DOAP vocabulary is in Annex A - The DOAP Vocabulary. The DOAP vocabulary imports the FOAF vocabulary[15] and contains 7 classes:

- Project - describes a project. The project class has 2 superclasses defined in other ontologies:
    - http://xmlns.com/wordnet/1.6/Project
    - http://xmlns.com/foaf/0.1/Project
- Version - provides information about a version of a project;
- Repository - gives information about the source code repository of a project. The repository class has 4 subclasses, for different kinds of repositories:
    - SVNRepository - a Subversion repository;
    - BKRepository - a BitKeeper repository;
    - CVSRepository - a CVS repository; and
    - ArchRepository - a GNU Arch repository.

The DOAP vocabulary has a number of properties, or relations. They are summarized in Table 2.

Table 2 - DOAP's Properties

Property	**Description**	**Domain**	**Range**
name[16]	A name of something.		Literal[17]
homepage[18]	URL of a project's homepage, associated with exactly one project.	Project	

[15] The FOAF vocabulary - http://xmlns.com/foaf/0.1/index.rdf - accessed: 16/06/2006.

[16] The “name” property is a rdfs:subPropertyOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#label".

[17] Literal = "http://www.w3.org/2000/01/rdf-schema#Literal".

[18] The “homepage” and “old-homepage” properties are OWL Functional Properties (rdf:type rdf:resource="http://www.w3.org/2002/07/owl#InverseFunctionalProperty") . They are also rdfs:subPropertyOf rdf:resource="http://xmlns.com/foaf/0.1/homepage" .

Property	Description	Domain	Range
old-homepage[18]	URL of a project's past homepage, associated with exactly one project.	Project	
created	Date when something was created, in YYYY-MM-DD format. e.g. 2004-04-05		Literal[17]
short_desc	Short (8 or 9 words) plain text description of a project.		Literal[17]
description	Plain text description of a project, of 2-4 sentences in length.		Literal[17]
release	A project release.	Project	Version
mailing-list	Mailing list home page or email address.	Project	
category	A category of a project.	Project	
license	The URI of an RDF description of the license the software is distributed under		
repository	Source code repository.	Project	Repository
anon-root	Repository for anonymous access.	Repository	Literal[17]
browse	Web browser interface to repository.	Repository	
module[19]	Module name of a CVS, BitKeeper or Arch repository.	owl:unionOf CVSRepository ArchRepository BKRepository	
location	Location of a repository.	Repository	
download-page	Web page from which the project software can be downloaded.	Project	

[19] The "module" property does not apply to Subversion repositories as it can be seen by the definition of its domain.

Property	Description	Domain	Range
download-mirror	Mirror of software download web page.	Project	
revision	Revision identifier of a software release.	Version	Literal[17]
file-release	URI of download associated with this release.	Version	
wiki	URL of Wiki for collaborative discussion of project.	Project	
bug-database	Bug tracker for a project.	Project	
screenshots	Web page with screenshots of project.	Project	
maintainer	Maintainer of a project, a project leader.	Project	Person[20]
developer	Developer of software for the project.	Project	Person[20]
documenter	Contributor of documentation to the project.	Project	Person[20]
translator	Contributor of translations to the project.	Project	Person[20]
tester	A tester or other quality control contributor.	Project	Person[20]
helper	Project contributor.	Project	Person[20]
programmin g-language	Programming language a project is implemented in or intended for use with.	Project	Literal[17]
os	Operating system that a project is limited to. Omit this property if the project is not OS-specific.	Project	Literal[17]

At the time of writing this work, there were two Web applications (DOAP A Matic[21] and DOAP-a-matic[22]) for the construction of DOAP files. The applications were not up-

20 Person = "http://xmlns.com/foaf/0.1/Person".

21 DOAP A Matic - http://crschmidt.net/semweb/doapamatic/ - accessed: 16/06/2006

to-date with the recent schema. Even though, the obligatory items were covered. Two other applications (DOAP embedded in .NET assemblies[23] and DOAPamine[24]) offered the possibility to describe projects while developing them. The DOAPamine application was up-to-date with the recent vocabulary; however, the update process seemed to be manual. That is, once the vocabulary changed, the developer changed the application. However, the DOAP vocabulary, and consequently the applications, did not cover all the requirements proposed by the SWC. Therefore, we created an extended DOAP vocabulary that we introduce in the next section.

3.3.3 The Extended DOAP Vocabulary

We reviewed the SWAPps submitted to the SWC in terms of an extended DOAP vocabulary. We extended the DOAP vocabulary in order to provide some other characteristics related to the challenge and to our objective of having at the end of this work a framework for SWAPps. The new characteristics are:

- The minimal and desirable requirements as presented by SWC's definition of a SWAPp [Visser & Klein, 2005]. This will also be done because, as cited in the minutes [W3C, 2005c], the challenge's definition is presented as a potential subset of the ADTF criteria;
- Some other characteristics, especially technological ones, will be included because they are of our interest to develop the framework.

The extended DOAP vocabulary, SWDOAP, is in Appendix 1 - The SWDOAP Vocabulary. In new vocabulary, we defined some new classes:

- Category - provides information about a category from a classification and categorization system;
- DescriptionLanguage - gives information about an ontology description language;
- DistributionMethod - a distribution method.
- Ontology - an ontology;
- PersistenceTech - a persistence technology;
- QueryDescriptionLanguage - an ontology query description language;
- ReasoningTech - a reasoning technology;
- SoftwareComponentType - The type of a software component. For example: agent, component;
- SupportingTech - a supporting technology.

Those new classes are used in conjunction with DOAP's ones in order to define the new characteristics of SWDOAP. For clarity, we grouped the new characteristics and those defined in DOAP in four aspects in the review of the SWC applications:

- metadata (about the application);

---

[22] DOAP-a-matic - http://www.bonjourlesmouettes.org/doapy/doap-a-matic.php.en - accessed: 16/11/2005 on Google's cache.

[23] DOAP embedded in .NET assemblies - http://usefulinc.com/doap/news/contents/2004/08-10-dotnet/read - accessed: 16/06/2006.

[24] DOAPamine - http://www.ontogon.com/doapamine/ - accessed: 16/06/2006

- data meaning;
- information sources; and
- applications.

Next, we present in Table 3 all the properties of the extended DOAP. They include the DOAP properties and the properties defined in SWDOAP, presented in italic. The properties are grouped by the four aspects.

Table 3 - Extended DOAP's Properties

Metadata Aspect			
Property	**Description**	**Domain**	**Range**
name[25]	A name of something.		Literal[17]
homepage[26]	URL of a project's homepage, associated with exactly one project.	Project	
old-homepage[18]	URL of a project's past homepage, associated with exactly one project.	Project	
created	Date when something was created, in YYYY-MM-DD format. e.g. 2004-04-05		Literal[17]

[25] The "name" property is a rdfs:subPropertyOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#label".

[26] The "homepage" and "old-homepage" properties are OWL Functional Properties (rdf:type rdf:resource="http://www.w3.org/2002/07/owl#InverseFunctionalProperty") . They are also rdfs:subPropertyOf rdf:resource="http://xmlns.com/foaf/0.1/homepage" .

Property	Description	Domain	Range
short_desc	Short (8 or 9 words) plain text description of a project.		Literal[17]
description	Plain text description of a project, of 2-4 sentences in length.		Literal[17]
release	A project release.	Project	Version
mailing-list	Mailing list home page or email address.	Project	
category[27,32]	A category of project.	Project	
license[28, 31]	The URI of an RDF description of the license the software is distributed under		
repository	Source code repository.	Project	Repository
anon-root	Repository for anonymous access.	Repository	Literal[17]
browse[30]	Web browser interface to repository.	Repository	
module[29]	Module name of a CVS, BitKeeper or Arch repository.	owl:unionOf CVSRepository ArchRepository BKRepository	
location	Location of a repository.	Repository	
download-page [32, 30]	Web page from which the project software can be downloaded.	Project	
download-mirror[30]	Mirror of software download web page.	Project	

[27] We could redefine this property, changing the range to SWDOAP:Category, but we preferred not do that in order to be compliant with DOAP. But we are going to use a SWDOAP:Category as a filler for this property.

[28] At a first glance, we thought that it would be better to define the domain of this property as doap:Project. However, to keep compliant with DOAP and considering that not only a Project ha a license, we did not change the domain of this property.

[29] The "module" property does not apply to Subversion repositories as it can be seen by the definition of its domain.

[30] As the description of this property suggests, its range is a homepage. However, its range is not defined as so. We could redefine this property making it an rdfs:subPropertyOf rdf:resource="http://xmlns.com/foaf/0.1/homepage", but we did not do that in order to be compliant with DOAP. Therefore, we are going to use a foaf:Document as a filler for this property.

Property	Description	Domain	Range
revision	Revision identifier of a software release.	Version	Literal[17]
file-release	URI of download associated with this release.	Version	
wiki[30]	URL of Wiki for collaborative discussion of project.	Project	
bug-database	Bug tracker for a project.	Project	
screenshots [30]	Web page with screenshots of project.	Project	
maintainer	Maintainer of a project, a project leader.	Project	Person[20]
developer	Developer of software for the project.	Project	Person[20]
documenter	Contributor of documentation to the project.	Project	Person[20]
translator	Contributor of translations to the project.	Project	Person[20]
tester	A tester or other quality control contributor.	Project	Person[20]
helper	Project contributor.	Project	Person[20]
affiliation	*The affiliation of a Project.*	*Project*	
metadata-observation	*Observation about the metadata about this project.*	*Project*	
last-visited	*Date of the last visit to the homepage of a project, in YYYY-MM-DD format. e.g. 2004-04-05.*	*Project*	*Literal*[17]
doap-url[31]	*The DOAP URL of a project.*	*Project*	
challenge-ranking	*Ranking reached by a project in the Semantic Web Challenge*	*Project*	
contact	*A contact of a project*	*Project*	*Person*[20]

[31] Maybe, the "best" range for this property would be a foaf:Document, but we did not define that in order to be compliant with DOAP. But as the range is not defined, we are going to use a foaf:Document as a filler for this property.

Property	Description	Domain	Range
challenge-year	*Year of the submission of the project to the Semantic Web Challenge.*	*Project*	
Data Meaning Aspect			
Property	**Description**	**Domain**	**Range**
ontology[32]	*An ontology used by a project.*	*Project*	*Ontology*
descriptionLanguage[32]	*An ontology description language used by a project.*	*Project*	*Description Language*
data-meaning-observation	*Observation about the use of data meaning done by the project.*	*Project*	
queryDescriptionLanguage[32]	*An ontology query description language used by a project.*	*Project*	*QueryDescriptionLanguage*
reasoningTech[32]	*A reasoning technology used by a project.*	*Project*	*ReasoningTech*
Information Sources Aspect			
Property	**Description**	**Domain**	**Range**
same-purpose-as-original[33]	*Is the data, manipulated by the project, used in a different purpose than original?*	*Project*	
information-sources-observation	*Observation about the information sources used by the project.*	*Project*	
structurally-heterogenous	*Does the project organize information in different ways?*	*Project*	
real-world-data [33]	*Does the project use real world data?*	*Project*	
audience-type	*Whom are the final users?*	*Project*	

[32] During the reviewing process, we felt the necessity to write an observation about this property. In order to do that, we used the reification mechanism of RDF. Therefore we assert that an annotation of the type http://www.w3.org/2000/10/annotationType#Comment annotates (http://www.w3.org/2000/10/annotation-ns#annotates) a triple that uses this property.

[33] A more formal definition of this requirement, by the SWC organizers, including metrics that could be used to evaluate this characteristic of the submitted applications would help to better describe this property.

Property	Description	Domain	Range
syntatically-heterogenous	*Does the project use different syntactic standards?*	*Project*	
diverse-ownership[32]	*Do the information sources of this project have diverse ownership?*	*Project*	
persistenceTech[32]	*A persistence technology used by a project.*	*Project*	
multiple-language[32]	*Do the information sources of this project support multiple languages?*	*Project*	
semantically-heterogenous	*Does the project use different terminologies to refer to the same information?*	*Project*	
data-domain	*What is the domain of data?*	*Project*	
distributionMethod[32]	*A distribution method used by a project.*	*Project*	
distributed[32,33]	*Are the information sources of the project distributed?*	*Project*	
multimedia[34]	*Does the project use the content of multimedia documents?*	*Project*	
scalable[33]	*How many data sources are used?*	*Project*	
diverse-method-of-access[32]	*Does the project support diverse methods of access? For example, mobile access.*	*Project*	
Applications Aspect			
Property	**Description**	**Domain**	**Range**
programming-language[32, 35, 36]	Programming language a project is implemented in or intended for use with.	Project	Literal[17]

[34] The SWC organizers do not mention if the multimedia content (or documents) has to be used "semantically" or not.

[35] We could redefine this property in order to have its range as empty, and then use a more appropriate class as filler. However, we did not do that in order to be compliant with DOAP.

[36] We use this property to define the programming language that project is implemented in. That is, we are not concerned about the programming language that the project is intended for use with.

Property	Description	Domain	Range
os[35]	Operating system that a project is limited to. Omit this property if the project is not OS-specific.	Project	Literal[17]
scalable-in-number-of-components[33]	*Is the project scalable in the number of components used?*	*Project*	
softwareComponentType[32]	*A software component type of a project*	*Project*	*SoftwareComponentType*
application-observation	*Observation about the applications aspect of the project.*	*Project*	
supporting-tech	*Supporting technology used by the project.*	*Project*	
open-source[32]	*By the DOAP definition, it was supposed to be a schema to describe open source projects. However, this is not the case for the projects of SWC. This property is the intended to explicit if a project is open source or not.*	*Project*	

The metadata (about the application) aspect provides information such as where on the Web we found information about that application, when its homepage was last visited, who are the contacts etc. The name of this aspect may be misleading. We are not dealing with the metadata characteristics of the application; on the other hand, we are describing the metadata about the application.

The data meaning aspect presents information about which ontologies, ontology description languages, query languages among other characteristics the application uses.

The information sources aspect offers a view on the characteristics of the information sources used by the application, e.g., which is the domain; if multimedia documents are used; if the information sources have diverse ownership; which persistence technology is used; if multiple languages are supported; if access through multiple devices is supported etc.

Finally, the applications aspect provides information on how the application was implemented, which programming language was used; which kind of software components were used; if the application is open source; what are the supporting technologies used by the application etc.

3.4 Summary

In this chapter, we presented the Semantic Web Challenge, its requirements and desirable qualities for Semantic Web applications. We also presented that the challenge ranks its applications but it is not concerned about their classification. We elicited some bibliography on classifying Semantic Web applications or ontology based applications.

In addition, we restricted our proposal of a Semantic Web application framework to be based on the applications submitted to the challenge. This restriction was reached through a domain analysis of the applications, shown in the next chapter. To perform the domain analysis we defined an extended DOAP vocabulary - SWDOAP. This vocabulary is one of the contributions of this work since it extends the DOAP vocabulary taking into account the requirements of the challenge. After Chapter 4 , containing the domain analysis of the applications, the following chapters (5 , 6 and 7) describe, in natural language, the applications grouped by the year of their submission.

4 SWC Applications Domain Analysis

We reviewed the 35 applications submitted to the SWC in order to develop a Semantic Web application framework. This revision, segmented by yearly edition of the challenge, is described in Chapters 5 , 6 and 7 . In the current chapter, we present the common functionalities of the applications[37], the types of application and the types of integration used by the applications described later in their respective chapter.

The review of the applications submitted to SWC was done using SWDOAP, already presented in Chapter 3 . We searched on the literature and on the internet for resources about the applications. Those were the main sources for capturing the information about the applications. We could have used others, for example, source code or applications' usage. However, these sources were not available for all the applications; therefore, we preferred to maintain our focus on the papers and homepages of the applications.

While looking for the information for "documenting" the applications, using SWDOAP, we learned about them and their functionalities. It was possible to perceive some "commons" functionalities offered. In addition, we also perceived some "similar" types of applications. Besides that, and because of some requirements from the SWC, we paid attention to the forms of integration of data done by the applications. We chose to present those characteristics in this chapter in an attempt not to be repetitive when describing the applications in their respective chapter.

Table 4, Table 5 and Table 6 present a summary of the common functionalities, types of application and types of integration that emerged from the review of the applications and how often they occurred along the years. The next subsections presents, for each functionality, type of application and type of integration, its definition and which are the applications that represent it.

[37] Actually only 25 applications were considered in the domain analysis. The remaining applications were not considered since: (a) there were not enough information available to review the application; (b) as the SWC's organizers recognized, in 2004, some applications [Klein & Visser, 2005] are "infrastructure applications" because they do not provide functionalities to the end user.

Table 4 - Functionalities Summary

Functionalities		Number of applications			
		2003	2004	2005	Total
1	Browse Functionality	6	9	4	19
2	Generation of Navigational Views Functionality	1	2	1	4
3	Dynamic and Semantic Linking Hypertext Structures Functionality	1	1	0	2
4	Search Functionality	7	9	3	19
5	Semantic Search Functionality	6	4	2	12
6	Semantic Query Expansion Functionality	2	3	1	6
7	Access through Diverse Devices Functionality	0	3	1	4
8	Support for Diverse Languages Functionality	2	2	1	5
9	Use of Multimedia Documents Functionality				
10	Multimedia Handling Functionality	0	1	0	1
11	Multimedia Metadata Functionality	3	1	1	5
12	Multimedia Generation Functionality	3	2	1	6
13	Semantic Growth Functionality	1	2	1	4
14	Semantic Recommender Policy Functionality	1	2	0	3
15	Ontology Functionality				
16	Ontology Schema Editor Functionality	0	2	1	3
17	Ontology Instances Editor Functionality	1	5	3	9
18	Ontology Repository Functionality	1	5	3	9

Table 5 - Types of Application Summary

Types of Application		Number of applications			
		2003	2004	2005	Total
1	Portal	6	9	4	19
2	Ontology Tool	0	2	1	3
3	Instance of a Framework	3	4	2	9
4	Semantic P2P Application	0	2	1	3
5	Semantic Collaborative Tool	2	0	0	2
6	Semantic Wiki	0	1	0	1

Table 6 - Types of Integration Summary

Types of Integration		Number of applications			
		2003	2004	2005	Total
1	Wrappers and Mediators Integration Functionality	7	8	4	19
2	Manual Integration Functionality	1	3	1	5

From Section 4.1 through Section 4.8 we define the functionalities of the applications submitted to the SWC that caught our eyes by their use of semantic or by the frequency they appeared. We also list the applications that use each one of the functionalities. More information about the applications can be found on Chapters 5 , 6 and 7 .

Section 4.9 presents the most common types of application that were submitted to the challenge and their respective applications. Additionally, Section 4.10 shows the two types of integration functionality that were most used by the applications and lists the applications that used each type.

4.1 Browse Functionality

An application offers the Browse functionality when the user is allowed to navigate the contents (or instances) of the knowledge base of the application. This occurs mainly through Web browsers, but other applications or interfaces can be used with the same purpose of traversing a set of linked objects.

Examples of applications that offer this functionality are:

- SEmantic portAL (SEAL) [Hartmann & Sure, 2004];
- Drug Ontology Project for Elsevier (DOPE) [Stuckenschmidt *et al.*, 2004];
- SEmantic COllaboration (SECO) [Harth, 2004];
- Building Finder [Michalowski *et al.*, 2004];
- CS AKTive Space [Shadbolt *et al.*, 2004];
- GeoShare [Hübner *et al.*, 2004];
- MusiDB [Stegers *et al.*, 2006];
- The Multilingual Access to Data Infrastructures of the European Research Area (MADIERA) Portal [Alvheim & Ryssevik, 2005];
- SemanticOrganizer [Keller *et al.*, 2004];
- Platypus Wiki [Tazzoli *et al.*, 2004];
- MuseumFinland [Hyvönen *et al.*, 2005];
- Semantic Portal of International Affairs (SPIA) [Contreras *et al.*, 2004];
- Flink [Mika, 2005a];
- Bibster [Haase *et al.*, 2004];
- Mediator EnvirOnment for Multiple Information Sources (MOMIS) [Beneventano & Bergamaschi, 2004];
- DynamicView [Gao *et al.*, 2005];
- Personal Publication Reader (PPR) [Baumgartner *et al.*, 2005];
- Oyster [Palma & Haase, 2005]; and
- CONFOTO [Nowack, 2005].

4.1.1 Generation of Navigational Views Functionality

Sometimes, the "data model" or ontology of an application is complex and the browsing experience can be overwhelming and frustrating for the end-user. Therefore, some applications also use an "extra" model to generate navigational views.

Examples of applications that offer this functionality are:

- SEmantic portAL (SEAL);
- MuseumFinland;
- Semantic Portal of International Affairs (SPIA); and
- Personal Publication Reader (PPR).

4.1.2 Dynamic and Semantic Linking Hypertext Structures Functionality

The Dynamic and Semantic Linking Hypertext Structures functionality only appeared in two applications. Nevertheless, in the case of GOHSE, its use of a proxy to enrich hypertext structures with semantic related data seemed promising for two reasons. The first is the dynamic aspect by the use of a proxy. The second is the semantic aspect, in which the data that goes through the proxy is indexed and matched "against" an integrated ontology on a specific domain to provide new links on the "old" hypertext structure.

Examples of applications that offer this functionality are:

- Annotated Terrestrial Information (AnnoTerra) [Ramagem *et al.*, 2004]; and
- GOHSE [Bechhofer *et al.*, 2005].

4.2 Search Functionality

When an application offers the Search functionality, the user can provide specific criteria and then the application will return the items in the knowledge base that match those criteria. The search functionality is a keyword search. For this kind of search, there are many well-established algorithms and even services on the Web. Nevertheless, for the user, having this kind of functionality is important to facilitate the knowledge base usage.

Examples of applications that offer the Search functionality are:

- SEmantic portAL (SEAL);
- Drug Ontology Project for Elsevier (DOPE);
- SEmantic COllaboration (SECO);
- Annotated Terrestrial Information (AnnoTerra);
- Building Finder;
- CS AKTive Space;
- GeoShare;
- MusiDB;

- The Multilingual Access to Data Infrastructures of the European Research Area (MADIERA) Portal;
- SemanticOrganizer;
- Platypus Wiki;
- MuseumFinland;
- Semantic Portal of International Affairs (SPIA);
- Unspecified Ontology (UNSO) [Ben-Asher & Berkovsky, 2004];
- Bibster;
- Mediator EnvirOnment for Multiple Information Sources (MOMIS);
- DynamicView;
- Oyster; and
- CONFOTO.

4.2.1 Semantic Search Functionality

Semantic Search functionality is a search functionality that takes into account the ontology (schema) that defines the knowledge base (instances) of an application. This can be done in several ways, for example, the application may let the user choose which concept she is looking for, or the application may do a syntactic match of the user query against the ontology and then offer options of broader or narrower concepts for the user to search. For example, in the Semantic Portal of International Affairs (SPIA), their Semantic Search Engine answers queries, posed in natural language or in forms, with instances instead of documents.

Examples of applications that offer this functionality are:

- SEmantic portAL (SEAL);
- Drug Ontology Project for Elsevier (DOPE);
- Annotated Terrestrial Information (AnnoTerra);
- Building Finder;
- CS AKTive Space;
- GeoShare;
- The Multilingual Access to Data Infrastructures of the European Research Area (MADIERA) Portal;
- Platypus Wiki;
- MuseumFinland;
- Semantic Portal of International Affairs (SPIA);
- DynamicView; and
- FungalWeb [Shaban-Nejad *et al.*, 2004] [Shaban-Nejad *et al.*, 2005].

4.2.2 Semantic Query Expansion Functionality

Some applications offer the possibility of expanding queries based on the ontology used by the knowledge base. We classified that functionality as a "Semantic Query Expansion Functionality".

Examples of applications that we considered as offering Semantic Query Expansion functionality are:

- Drug Ontology Project for Elsevier (DOPE);
- GeoShare;
- MusiDB;
- Unspecified Ontology (UNSO);
- GOHSE; and
- Personal Publication Reader (PPR).

The difference between the Semantic Search Functionality and the Semantic Query Expansion Functionality is that the latter may be applied to the former in order to narrow or broaden the search.

4.3 Access through Diverse Devices Functionality

A desirable quality for the SWC is that users should be able to access the applications through devices other than a personal computer, the applications that offered this functionality are:

- MuseumFinland;
- Semantic Portal of International Affairs (SPIA);
- Annotea Shared Bookmarks [Koivunen, 2005]; and
- CONFOTO.

The access to data through different devices can be at same time a facilitator for the user as well as a way of providing accessibility to those with special needs. For example, a user can "present" some disabilities in different contexts such as noisy and slow Internet connection environments. Those disabilities may be compared to those of a person with special needs. Therefore, the support to access through diverse devices can improve people's lives and raise their standard of living [Seeman, 2004]. International guidelines have already been standardized and recommended on how to implement those accessibilities for the Web (Web Content Accessibility Guidelines[38] - WCAG).

4.4 Support for Diverse Languages Functionality

Another desirable quality for the Semantic Web Challenge is that users should be able to access the applications in multiple languages. Among the applications reviewed, the following ones presented that functionality:

---

[38] WCAG - http://www.w3.org/WAI/intro/wcag.php - accessed: 16/06/2006

- SEmantic portAL (SEAL);
- CS AKTive Space;
- The Multilingual Access to Data Infrastructures of the European Research Area (MADIERA) Portal;
- Semantic Portal of International Affairs (SPIA); and
- DynamicView.

4.5 Use of Multimedia Documents Functionality

The SWC defines that a desirable quality for the SWAPps is the use of the contents of multimedia documents. In reviewing the applications submitted to challenge, we found three manners in which applications can do that. Those manners are explained in the next sub-sections.

4.5.1 Multimedia Handling Functionality

The application can handle multimedia documents, but it does not make any use of the metadata about the multimedia document. For example, SemanticOrganizer can store attachments from e-mails, but it is concerned with the metadata provided by the e-mail, not by the attachment.

4.5.2 Multimedia Metadata Functionality

In this functionality, the application acquires, stores and uses metadata about multimedia documents. Among the applications reviewed, the following presented that functionality:

- Building Finder;
- CS AKTive Space;
- GeoShare;
- MuseumFinland; and
- CONFOTO.

4.5.3 Multimedia Generation Functionality

In this functionality, the application is able to generate multimedia documents (data or metadata) from its knowledge base. For example, an application is able to generate a map that points researchers locations based on their addresses (CS AKTive Space, Flink).

Examples of applications that offer this functionality are:

- Drug Ontology Project for Elsevier (DOPE);
- CS AKTive Space;
- GeoShare;
- Semantic Portal of International Affairs (SPIA);

- Flink; and
- DynamicView.

4.6 Semantic Growth Functionality

This functionality allows the knowledge bases to grow based on what is already stated on them. That kind of growth can happen in many ways. One way that a knowledge base could grow is based on inferences. In this specific case, it happens based on an inference mechanism, which is the identification of a resource that is represented in several sources but the information is incomplete in some of them.

Examples of applications that offer that functionality are:

- CS AKTive Space;
- SemanticOrganizer;
- Bibster; and
- Oyster.

4.7 Semantic Recommender Policy Functionality

Applications that offer Semantic Recommender Policy functionality are able to recommend to their users metadata that match the users' interests or profile. The data recommended can be found based on the similarity between users' profiles or on the knowledge base facts.

In Personal Publication Reader (PPR), reasoning is performed over collected semantic descriptions and additional knowledge bases like ontologies and user profile information.

Examples of applications that offer this functionality are:

- Semblog [Takeda & Ohmukai, 2005];
- MusiDB; and
- MuseumFinland.

4.8 Ontology Functionality

The ontology functionality is offered by applications that are able to handle metadata represented in the form of an ontology. This functionality can be composed of three different sub-functionalities explained in the next sections. The three different sub-functionalities are not an exhaustive list of all possible functionalities dealing with ontologies; we could have, for example, functionalities for aligning, merging, versioning and analyzing ontologies. The three sub-functionalities presented were the most commonly found on the applications.

4.8.1 Ontology Schema Editor Functionality

An application that offers the Ontology Schema Editor functionality is able to change the schema of an ontology, but it is not able to edit the instances of that ontology.

In a "pure Description Logics world", a more appropriate name would be a TBox Editor, but, in this work, the definition of ontology is broad-ranging (see Section 2.1) and this is the reason to choose ontology schema instead of TBox.

Examples of applications that offer this functionality are:

- Platypus Wiki;
- Unspecified Ontology (UNSO); and
- Oyster.

4.8.2 Ontology Instances Editor Functionality

An application that offers an Ontology Instances Editor functionality is able to change the instances of an ontology, but it is not able to edit the schema of that ontology.

In a "pure Description Logics world", a more appropriate name would be an ABox Editor, in this work the definition of ontology is broad-ranging (see Section 2.1) and this is the reason to choose ontology instances instead of ABox.

Examples of applications that offer this functionality are:

- Semblog;
- SemanticOrganizer;
- Platypus Wiki;
- Unspecified Ontology (UNSO);
- Bibster;
- Annotea Shared Bookmarks;
- Personal Publication Reader (PPR);
- Oyster; and
- CONFOTO.

4.8.3 Ontology Repository Functionality

An application that offers an Ontology Repository functionality is able to store ontologies (schema and instances). Sometimes those applications also offer some inference and querying mechanisms.

Examples of applications that offer this functionality are:

- Semblog;
- SemanticOrganizer;
- Platypus Wiki;
- Unspecified Ontology (UNSO);
- Bibster;
- Annotea Shared Bookmarks;
- Personal Publication Reader (PPR);
- Oyster; and

- CONFOTO.

4.9 Types of Applications

While reviewing the applications, it was clear that we could not isolatedly considerate the functionalities that they offered. Sometimes the specific domain or area addressed by them was also important in their characterization. Within a specific domain or area, it is possible to "narrow the focus" and better evaluate the possible solutions in order to improve the quality of the software to be developed.

For the applications submitted to SWC, we found that the grouping of some functionalities, already presented, would represent a type of application, for example portals and ontology tools. Other applications were instances of frameworks that had a "broader context" than the applications themselves. We also identified some applications with common factors such as the kind of interaction between users and the application (semantic collaborative tools) and between instances of an application (semantic P2P applications) needed to be taken into account when handling metadata and providing the user with an attractiveness factor.

As in the case of the functionalities, this enumeration of types of applications is not intended to be exhaustive, as well an application may be classified in more than one type since the types are not intended to be exclusive. In the next sections, we present the most common types of applications that we found on the applications submitted to SWC.

4.9.1 Portal

When we say that application is a Portal, we mean that the application offers, at least, two functionalities: browsing and searching.

Examples of applications that we considered as portals are:

- SEmantic portAL (SEAL);
- Drug Ontology Project for Elsevier (DOPE);
- SEmantic COllaboration (SECO);
- Building Finder;
- CS AKTive Space;
- GeoShare;
- MusiDB;
- The Multilingual Access to Data Infrastructures of the European Research Area (MADIERA) Portal;
- SemanticOrganizer;
- Platypus Wiki;
- MuseumFinland;
- Semantic Portal of International Affairs (SPIA);
- Flink;
- Bibster;

- Mediator EnvirOnment for Multiple Information Sources (MOMIS);
- DynamicView;
- Personal Publication Reader (PPR);
- Oyster; and
- CONFOTO.

4.9.2 Ontology Tool

An ontology tool is an application able to edit an ontology schema and instances and to store that information. That is, the ontology tool offers, at least, the three sub-functionalities of the Ontology Functionality:

- Ontology Schema Editor Functionality;
- Ontology Instances Editor Functionality; and
- Ontology Repository Functionality.

Applications that we classified as ontology tools are:

- Unspecified Ontology (UNSO);
- Annotea Shared Bookmarks; and
- Oyster.

4.9.3 Instance of a Framework

Some requirements from the SWC, naturally, led a developer to think in terms of a framework because this approach is a way to create applications that are extensible and customizable. Examples of those requirements are the need to be scalable in number of components working together and the integration of two or more distinct data sources.

Some papers, describing the applications to the SWC, reveal only the framework part of their implementation. If we considered only that, they would fit in the Infrastructure application as defined by the organizers in 2004 [Klein & Visser, 2005]. However, we tried to keep our focus on the instances submitted and how their functionalities were used and implemented.

Other papers only focus on the framework instantiation and, sometimes, it is hard to understand the rationale of some decisions taken. However, the instantiation of frameworks is, in the vast majority, a way to reuse recognized solutions to issues such as ontology storage and inference mechanisms.

Examples of Instance of a Framework applications are:

- SEmantic portAL (SEAL);
- Drug Ontology Project for Elsevier (DOPE);
- Semblog;
- MuseumFinland;
- Bibster;

- Mediator EnvirOnment for Multiple Information Sources (MOMIS);
- GOHSE;
- Personal Publication Reader (PPR); and
- Oyster.

4.9.4 Semantic P2P Application

A peer-to-peer network is a network where there is not a notion of a server and clients. However, there is the notion of nodes that act as client and servers, the peers. P2P applications are used to share computing power or data without a central server. In this work, we consider a Semantic Peer-to-Peer application as an application that is able to exchange metadata described according to an ontology without a central server.

Examples of Semantic P2P applications are:

- Unspecified Ontology (UNSO);
- Bibster; and
- Oyster.

4.9.5 Semantic Collaborative Tool

A collaborative tool, in this work, is a tool where a group of users can manipulate a shared resource. In Computer Science, there are specific areas that study that kind of applications such as Computer Supported Cooperative Work [Greif, 1988] and Groupware [Ellis *et al.*, 1991]. However, it is not our intention to go deeper in the discussion of collaborative work here. We will try to keep our focus on the use of semantics by collaborative tools. Therefore, a Semantic Collaborative tool is a tool where a group of users can manipulate shared metadata.

While reviewing the applications, an issue that rose was the relation between a Semantic Collaborative tool and a Semantic P2P application. At first, they seemed intrinsically related, because there is metadata shared and manipulated by a group of users using those applications. However, if we consider a more "traditional" definition of P2P applications, data is not the only resource that could be shared by this kind of applications. Sometimes, they can share computing power or other devices. In addition, a collaborative tool does not, necessarily, have to be implemented in a P2P network; it can use, for example, a Client-Server architecture.

Examples of Semantic Collaborative Tools are:

- Semblog; and
- CS AKTive Space.

4.9.5.1 Semantic Wiki

Platypus Wiki is the representing application of this type. It uses the same approach of Wiki pages [Leuf & Cunningham, 2001] but in its case to enable the collaborative editing of vocabularies and ontologies according to RDF Schema and OWL.

4.10 Types of Integration

What seemed to us as one of the most relevant aspects of the minimal requirements in the definition of a SWAPp was the need to use distributed, non-controlled-ownership and heterogeneous information sources. Those requirements can be seen as a well-defined research field in Computer Science, more specifically, in the Database area: the integrated access to multiple databases.

However, sometimes the semantic or formal description of the data meaning is embedded on the schema of databases and not explicit as in the case of the Semantic Web. On the Semantic Web, the explicitness of the data meaning is represented by the ontologies' schema and, therefore, it must be considered as an important aspect too, as it is by the SWC minimal requirements.

In the Database area, the approaches for integrated access to multiple databases can be, generally, classified into two categories: schema integration and the federated approach [Fileto, 2003]. The majority of the applications reviewed use the schema integration approach to integrate the data sources. Only one of the reviewed applications uses an approach similar to a federated approach to integrate the data. Some types of functionalities, discussed in the next sub-sections, were used to integrate the data: wrappers and mediators; and manual integration. There are other types of integration that could have been used but there are no such cases among the SWC applications.

4.10.1 Wrappers and Mediators Integration Functionality

In this functionality, two types of components are used to offer an integrated view of multiple information sources. A wrapper encapsulates each information source and communicates with the mediator. The mediator is responsible for offering a common schema for data access to the information sources through the wrappers or other mediators. That separation of mediators and wrappers, sometimes, is not so clear. There are mediators that have built-in wrappers in them. Moreover, there are wrappers that work as mediators as well. The important aspect of this functionality is that the application access a "common" integrated schema that is followed by the mediator and consequently by the wrappers.

Examples of applications that use this Integration Functionality:

- SEmantic portAL (SEAL);
- Drug Ontology Project for Elsevier (DOPE);
- SEmantic COllaboration (SECO);
- Annotated Terrestrial Information (AnnoTerra);
- Building Finder;
- CS AKTive Space;
- GeoShare;
- MusiDB;
- The Multilingual Access to Data Infrastructures of the European Research Area (MADIERA) Portal;
- MuseumFinland;

- Semantic Portal of International Affairs (SPIA);
- Flink;
- Bibster;
- Mediator EnvirOnment for Multiple Information Sources (MOMIS);
- GOHSE;
- DynamicView;
- Personal Publication Reader (PPR);
- Oyster; and
- CONFOTO.

4.10.2 Manual Integration Functionality

In this functionality, the integration of data is predefined by human interference. Examples of applications that use this integration functionality are:

- Semblog;
- SemanticOrganizer;
- Platypus Wiki;
- Annotea Shared Bookmarks; and
- FungalWeb.

4.11 Summary

We observed that a few functionalities were used by most of the applications, e.g., Browse, Search and Semantic Search functionalities. Most of the applications are Portals or Instances of a Framework and use the Wrappers and Mediators Integration functionality.

This chapter offered an overview of the domain analysis carried out on the applications submitted to SWC detailed in the next three chapters. This domain analysis is a contribution of this work since it will be used as a resource for the definition a Semantic Web application framework. It is also important to remark that the use of the SWC submissions was a choice made in this work as explained in Section 1.2 , but we could have used other sets of applications. For example, the one defined by the Applications and Demos Task Force[39] (ADTF) of the Semantic Web Best Practices and Deployment[40] (SWBPD) working group.

[39] ADTF - http://esw.w3.org/topic/SemanticWebBestPracticesTaskForceOnApplicationsAndDemos - accessed: 29/11/2005.

[40] SWBPD - http://www.w3.org/2001/sw/BestPractices/ - accessed: 29/11/2005.

After the elicitation of the functionalities, types of applications and integration methods we believe that some trends can be expected in the Semantic Web field. For example, interesting applications to users might be those that allow some form of social use, integrating blogs, wikis and personal data. The social use of the applications can facilitate the acquisition of data and metadata. Social networks also would allow some forms of validation of content, for example, by trusting in content recommended by a friend or co-worker. There are some applications “out there” already; however, they still lack a more formal semantics definition and the respective gain from its use. Other applicatlions that migth “catch” users are those that offer enhanced annotation techiniques, which can improve the amount of data available on the Semantic Web.

5 Semantic Web Challenge 2003 Applications

The additional goal defined to SWC 2003 was that the applications should integrate at least two heterogeneous XML data or information sources that the application's author did not manage and that allow different viewpoints. SWC 2003 had 10 submissions presented in the next sections and summarized in Section 5.11 (Table 7).

The Chairs of the challenge got some interesting conclusions [Klein & Visser, 2004]: the ontologies were quite straightforward; RDF Schema would provide the necessary support by itself, OWL's additional expressivity would not be necessary; a few ontologies had more than 100 concepts; most of the ontologies functioned as a schema for the data, and other as guides for the users in finding information.

5.1 SEmantic portAL (SEAL)

The core SEmantic portAL (SEAL) [Maedche *et al.*, 2002] approach aims to use ontologies to manage community Web sites and Web portals. The ontology permits queries to multiple sources. In addition, the use of schema information by SEAL allows automatic generation of navigational views. The submission to SWC 2003 was not the conceptual framework SEAL itself; it was an implementation and extension of SEAL. SEAL was improved with a framework for integrating knowledge which includes five conceptual layers as it can be seen in Figure 6 [Hartmann & Sure, 2004].

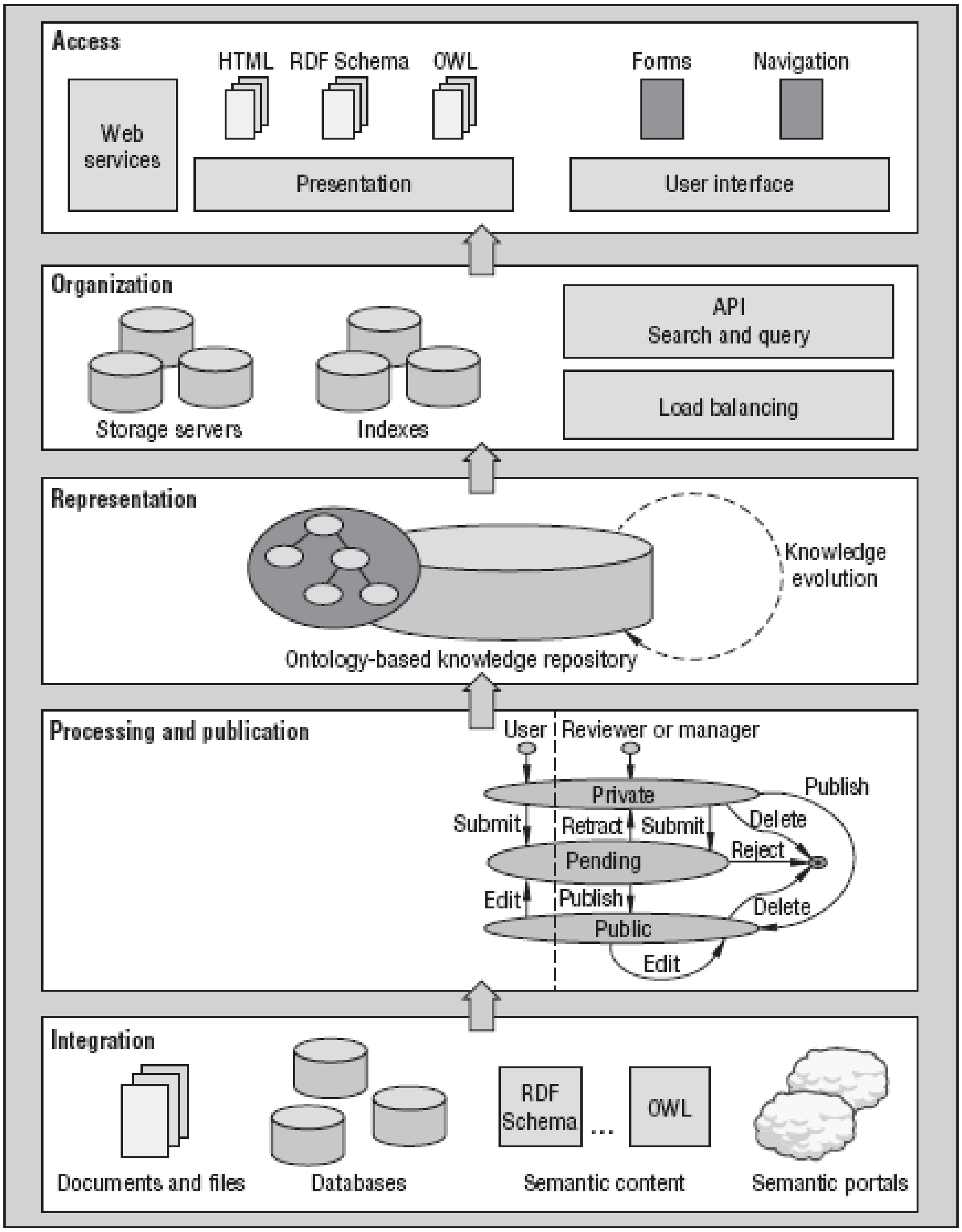

Figure 6 - The extended SEAL framework's five conceptual layers. [Hartmann & Sure, 2004]

The layers in Figure 6 can be seen as knowledge workflows, from the bottom layer (integration) to the top layer (access, for example, in a Web service).

The *integration layer* contains a set of modules, each able to handle a particular information source. The *processing and publication layer* creates content instances and provides a set of knowledge-processing methods. The *representation layer* uses ontologies and associated knowledge representation languages, such as RDF Schema and OWL to represent knowledge. The *organization layer* offers methods for indexing and search functionalities. At last, the *access layer* offers methods for showing content in several output formats and defines interaction interfaces [Hartmann & Sure, 2004].

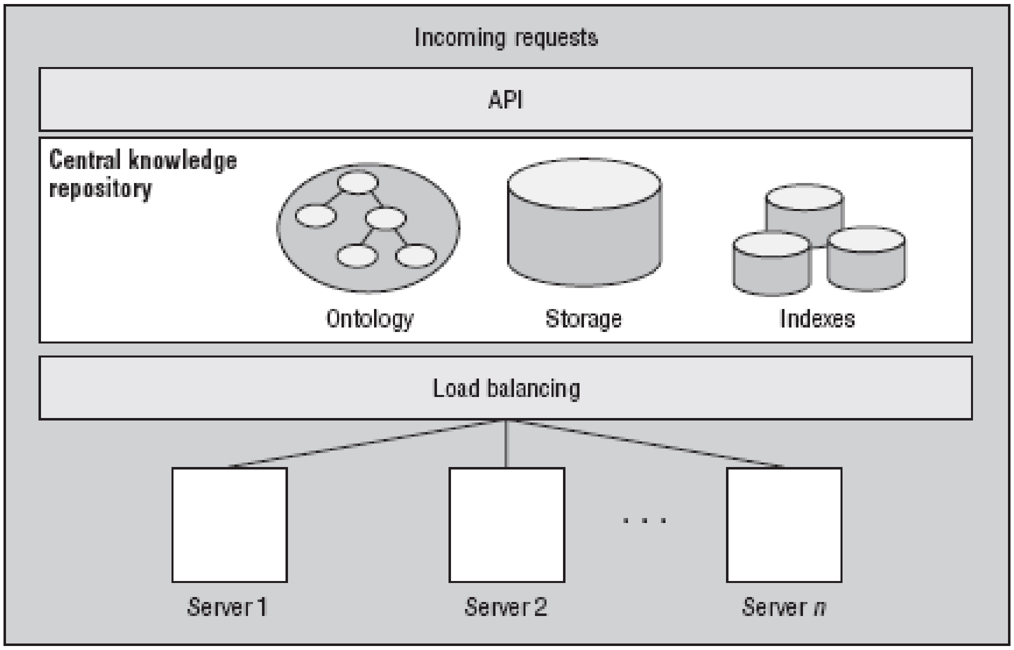

Figure 7 - SEAL Knowledge organization. [Hartmann & Sure, 2004]

As an example of the improvements in the core SEAL, in Figure 7, the architecture of a scalable storage mechanism to set up distributed servers in a cluster for handling several requests is presented.

The extended SEAL instance submitted to SWC was a Portal and an Instance of a Framework.

The extended SEAL uses a Wrappers and Mediators Integration functionality.

The elxtended SEAL offers the following functionalities:

- Browse functionality;
- Generation of Navigational Views functionality;
- Search functionality;
- Semantic Search functionality; and
- Support for Diverse Languages functionality;

5.2 Drug Ontology Project for Elsevier (DOPE)

The Drug Ontology Project for Elsevier (DOPE) provides access to multiple life science information sources through a single and innovative user interface. The interface relies on a thesaurus-based search system that was developed. The system uses automatic indexing, RDF-based querying, and concept-based visualization. [Stuckenschmidt *et al.*, 2004]

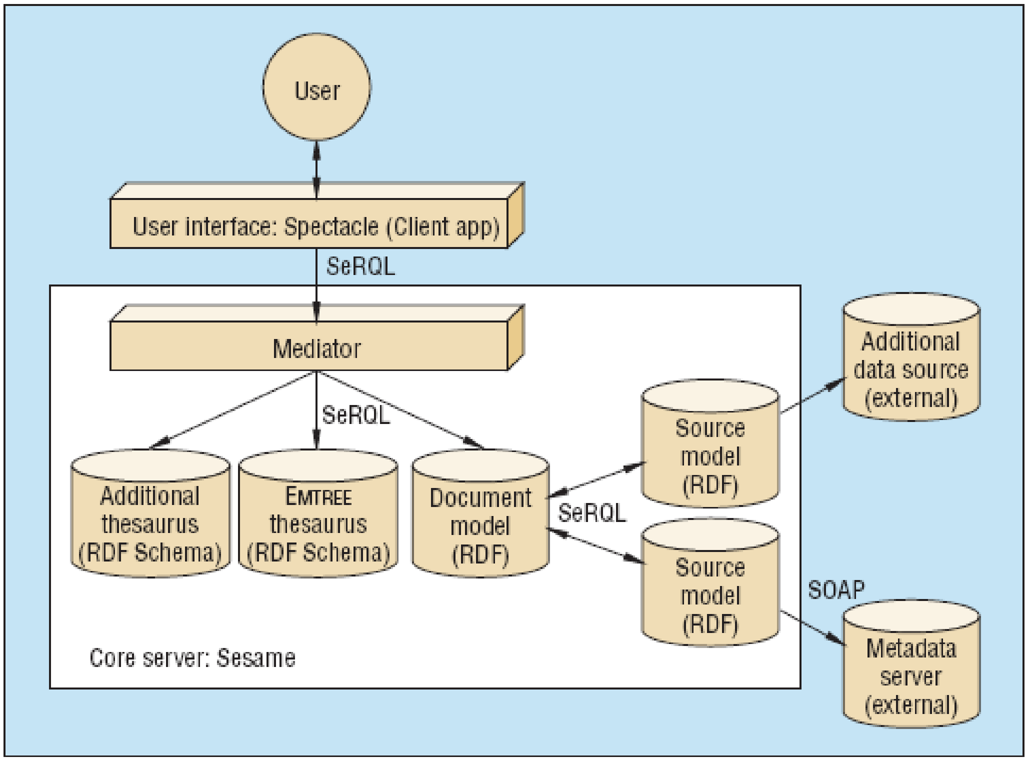

Figure 8 - Basic schematic of the DOPE architecture (protocols and data formats are in parentheses) [Stuckenschmidt *et al.*, 2004]

Figure 8 depicts DOPE's architecture. The *DOPE Browser* provides the user interface that guides users in exploring the information space and presents the query results in a structured way. The browser makes querying and navigation "transparent" to user, while abstracting the several data sources involved or the mappings used to integrate them.

The *DOPE Browser* uses Aduna's thesaurus-driven, interactive visualization technology, the Spectacle Cluster Map [Fluit *et al.*, 2005] for creating overviews and navigating the information. The user interface communicates, through the Sesame Query and Transformation Language (SeRQL) [Broekstra *et al.*, 2002], with an *infrastructure* to mediate between the information sources, thesaurus representation and external document metadata that was implemented using the RDF repository Sesame [Broekstra *et al.*, 2002].

DOPE is Portal. We also considered it as an Instance of Framework because it was implemented using Sesame.

DOPE uses a Wrappers and Mediators Integration functionality.

DOPE offers the following functionalities:

- Browse functionality;

- Search functionality;
- Semantic Search functionality;
- Semantic Query Expansion functionality; and
- Multimedia Generation functionality.

5.3 SEmantic COllaboration (SECO)

SEmantic COllaboration (SECO) is an infrastructure of mediators that allows agents to access data that is potentially spread across the Web [Harth, 2004].

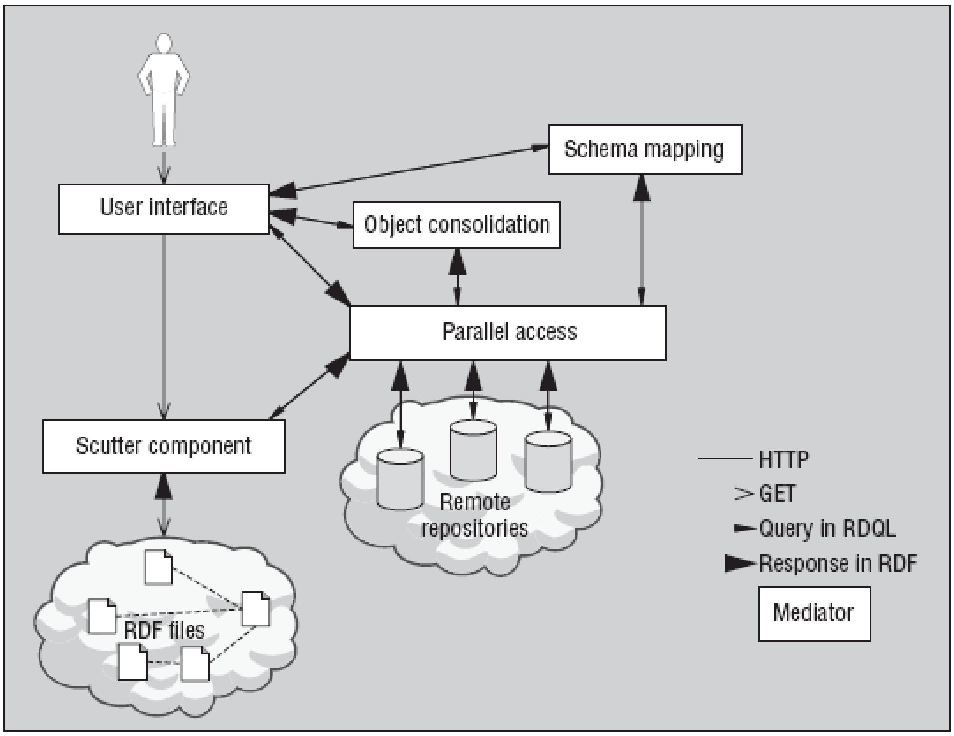

Figure 9 - SECO's Architecture [Harth, 2004].

In Figure 9 we present the architecture of SECO. SECO's components, or mediators, are interposed between databases and other information sources. Once all data sources are queryable, SECO obtains the needed data from repositories and integrates them on demand. The *scutter component* acts as an aggregation mediator that gathers RDF files from the Web, aggregates them, and enables software agents to query the RDF data set using a remote query interface. A mediator provides *parallel access* to all available data repositories. Other mediator services consolidate instances (*object consolidation*) and perform *schema mapping* based on an inference engine. The *interface* lets users browse the integrated data.

SECO is a Portal.

SECO uses a Wrappers and Mediators Integration functionality.

SECO offers the following functionalities:

- Browse functionality;
- Search functionality;

5.4 Annotated Terrestrial Information (AnnoTerra)

Annotated Terrestrial Information[41] (AnnoTerra) is a prototype knowledge-based system that uses Semantic Web technologies to offer enhanced earth science news feeds by doing focused semantic searches on NASA resource catalogs using earth science concepts and relationships. AnnoTerra users receive, as result, improved news feed with a list of system-determined data sets related to each news item. [Ramagem *et al.*, 2004].

Figure 10 illustrates AnnoTerra's components. They are:

- The Earth Observatory which provides news feeds;
- The Global Change Master Directory[42] (GMCD);
- Earth Observing System ClearingHouse (ECHO);
- An ontology for each of the previous components, respectively:
    - ♦ Earth science ontology;
    - ♦ GCMD DIF ontology; and
    - ♦ ECHO DIF ontology.
- The AnnoTerra component itself, which processes news feeds to extract meaningful keywords from textual information. It then performs semantic searches using those keywords, which are mapped to concepts in an ontology, in the GCMD for potentially relevant resources (for example, data set descriptions, images, documents etc.). The retrieved items are mapped from GCMD to ECHO using an ontology. The mapped items are then used to search ECHO's catalog for data sets. Consequently, the developers reached the integration of GCMD and ECHO through ontologies by asserting direct and indirect equivalencies between the concepts and data structures and between the data elements themselves.

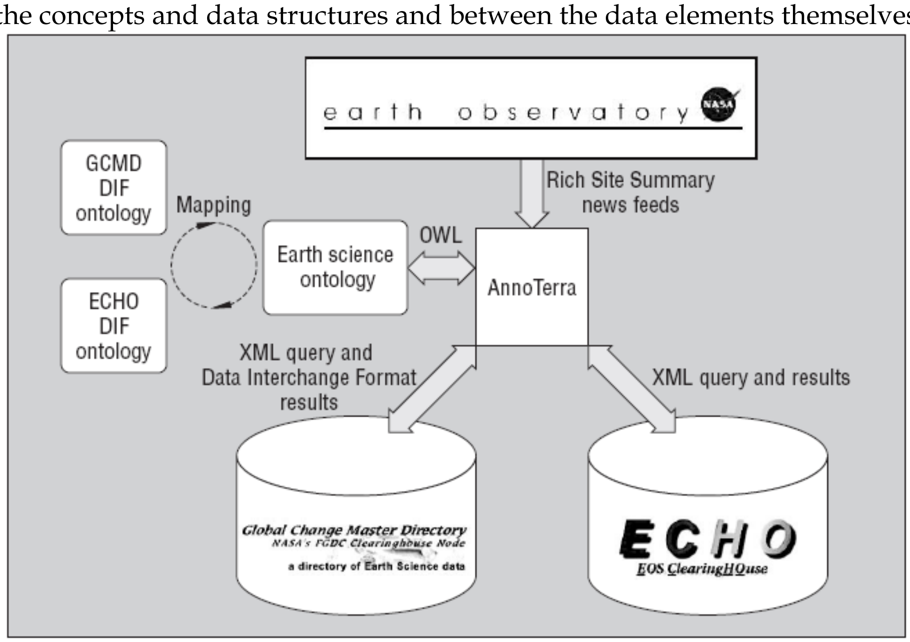

Figure 10 - AnnoTerra's components [Ramagem *et al.*, 2004].

41 AnnoTerra - http://annoterra.ssaihq.com/about.html - accessed 18/07/2005 on Google's cache.

42 GMCD - http://gcmd.nasa.gov/ - accessed: 16/08/2005

AnnoTerra uses a Wrappers and Mediators Integration functionality.
AnnoTerra offers the following functionalities:

- Dynamic and Semantic Linking Hypertext Structures functionality because it enhances the news feeds with items from some earth science catalogs; To offer this functionality, AnnoTerra uses:
  - ♦ Search functionality; and
  - ♦ Semantic Search functionality;

5.5 Building Finder

Building Finder combines satellite imagery, geospatial data, and structured and semi-structured data from diverse online data sources using Semantic Web technologies. Users can query an integrated view of these sources and request Building Finder to superimpose buildings and streets obtained from various sources on satellite imagery [Michalowski *et al.*, 2004]. Users can navigate through the Building Finder interface manually or have agents to query the application using RDF Data Query Language (RDQL).

To integrate semantically heterogeneous information from various data sources, Building Finder uses a number of technologies [Michalowski *et al.*, 2004]:

- Machine-learning techniques for converting traditional legacy Web sources and databases into Web services;
- A record linkage system for integrating data from various sources referring to a single entity;
- A mediator system providing uniform access to data from various Web services;
- An efficient execution system for information-gathering agents;
- RDQL and RDF formalisms for representing queries and query results respectively.

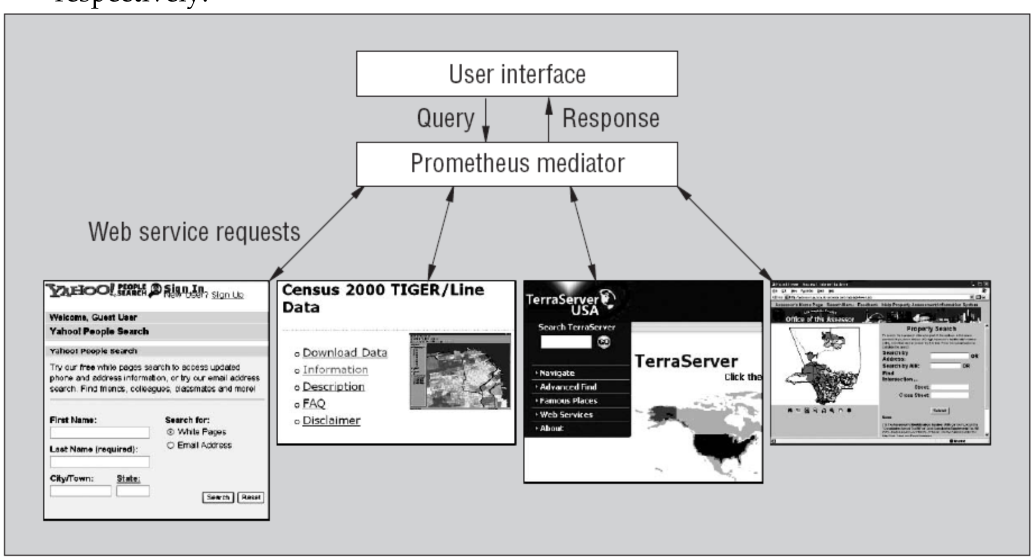

Figure 11 - Mediator execution [Michalowski *et al.*, 2004].

In Figure 11, it is possible to see a general representation of the Building Finder application. It is, in a simple way, composed by:

- A mediator: its goal is to provide unified access to diverse data sources;
- Data sources that are adapted into Web Services, using machine-learning techniques, by wrappers;
- A user interface.

Building Finder uses the Prometheus [Thakkar *et al.*, 2003] mediator. Prometheus runs over Theseus [Barish & Knoblock, 2002] to be able to consolidate the data sources. Theseus is an execution platform for information agents, which was augmented with underlying technologies initially developed for Active Atlas [Tejada *et al.*, 2002], a record-linkage system. Active Atlas compares objects' shared attributes to identify matching objects.

Prometheus mediator has three parts (Figure 12):

- A data model;
- A query reformulation component;
- A query execution component using the Datalog to Theseus converter.

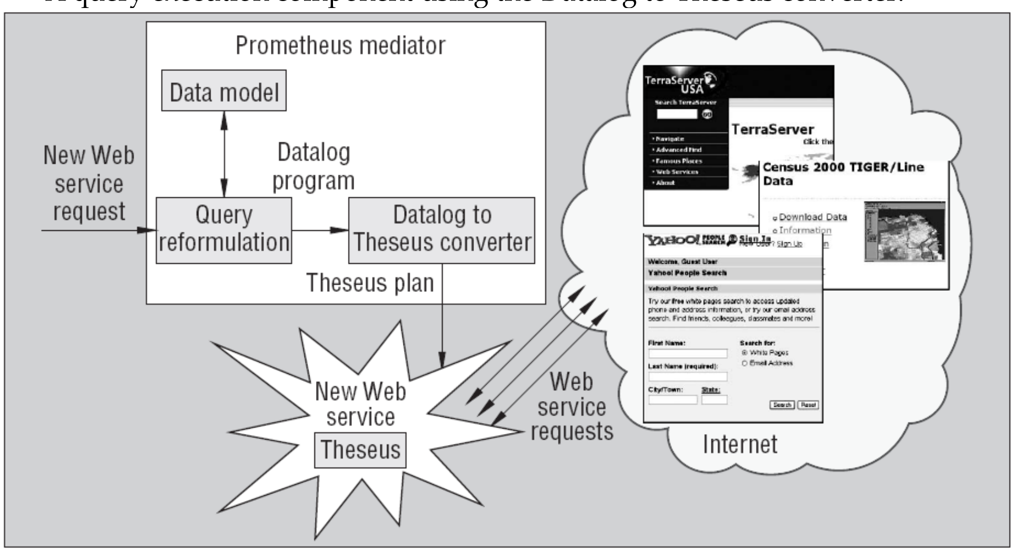

Figure 12 - Mediator architecture [Michalowski *et al.*, 2004].

The mediator recognizes queries on any arrangement of domain predicates. On receiving a query, the mediator combines it with the domain model to produce a Datalog program that can respond to the user query. The mediator then executes the produced program to find the results of the user query using the Theseus execution engine. For example, the Datalog converter receives a request in RDQL format from the user interface and converts it to a correspondent Datalog query. When it receives this query from the Datalog converter, the mediator uses its domain model to generate a Theseus plan to obtain data from Web sites. It then passes the generated plan to the Theseus execution engine. In Building Finder, Theseus provides efficient execution of mediator-created information-gathering plans [Michalowski *et al.*, 2004].

Resuming, Building Finder queries the mediator using RDQL queries, which are subsequently processed and executed. The mediator uses an internal RDQL to Datalog converter to interpret and process the query. On completing the query, the XML results produced by the mediator are converted to RDF and returned to the user.

Building Finder is a Portal.

Building Finder uses a Wrappers and Mediators Integration functionality.

Building Finder offers the following functionalities:

- Browse functionality;
- Search functionality;
- Semantic Search functionality; and
- Multimedia Metadata functionality.

5.6 Semblog

Semblog developers are not concerned only about information handling on the Web (collect, create and donate information). They are also interested in communication handling (relate, collaborate and present people). For that purpose, they go into the "Weblog field" trying to provide a smooth path using classic Web and Semantic Web technologies. For more information about Weblogs please refer to [Blood, 2002]. We detail some of the Weblog-specific terms used in this section in Chapter 10 (Glossary - Acronyms and Vocabulary).

Takeda and Ohmukai divided the architecture of Web systems in four layers (Figure 13 and Figure 14) from the metadata point of view:

- Format;
- Management;
- Aggregation; and
- Application [Takeda & Ohmukai, 2005].

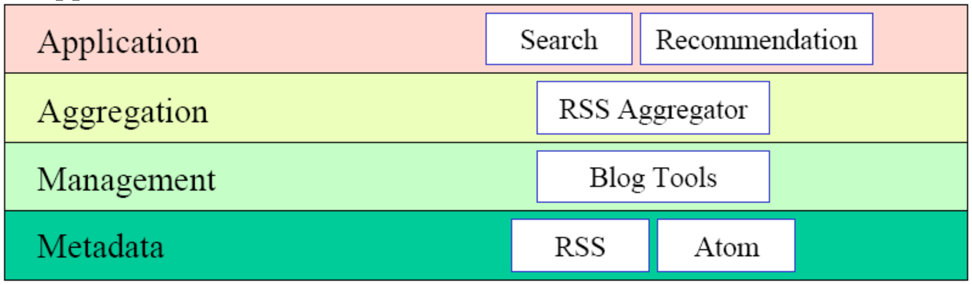

Figure 13 - Weblog Architecture [Takeda & Ohmukai, 2005].

In Figure 13, the developers of Semblog present the Weblog technologies and tools for each layer. In Figure 14, the developers show the semantically enhanced tools and technologies proposed by Semblog, as well a distinction between the content and social relationship aspects.

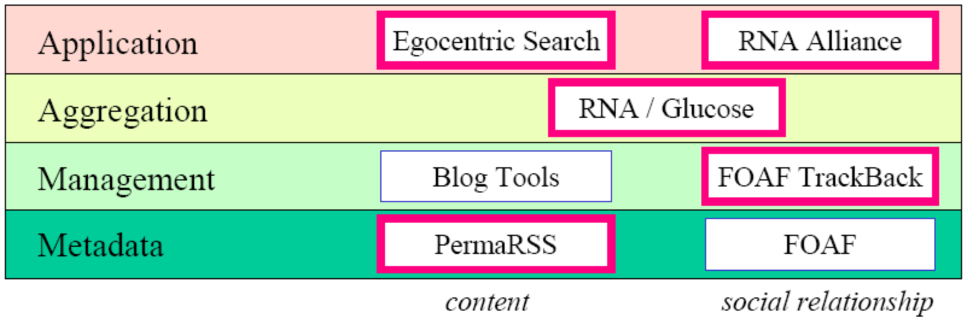

Figure 14 - Semblog Architecture [Takeda & Ohmukai, 2005].

Semblog is a suite that supports communication and information activities (Figure 15). The metadata about people and interpersonal relations plays a role on including activities on the communication level. The boxes with thicker borders, in Figure 14, are the ones proposed by the developers of the project. Those proposals are summarized bellow [Takeda & Ohmukai, 2005]:

- RNA is a Web-based RSS aggregator written with Perl;
- Glucose is also an RSS aggregator but a standalone application for Windows. It was developed to support information distribution process coordinating with RNA;
- RNA can generate FOAF data. RNA also has an interface for FOAF management to extend social network easily. This method is called "FOAF Track-Back" [Ohmukai *et al.*, 2004];
- RNA Alliance is a content recommendation system based on cooperation of multiple RNAs;
- Egocentric Search is a search that supports the authoring process by collecting related documents based on the user's representation as a network of documents, e.g., Weblog posts [Numa *et al.*, 2004].

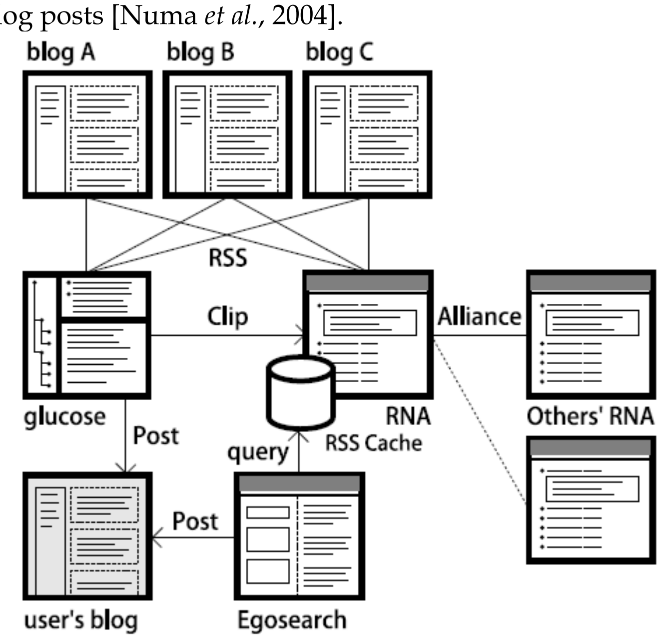

Figure 15 - Semblog's System Architecture [Ohmukai *et al.*, 2004]

Semblog offers a "value aggregation" to the Weblog field. It is not an ordinary Weblog, since, besides other improvements, it is concerned about the social relationships using FOAF.

Semblog is a framework, as defined by the authors [Takeda & Ohmukai, 2005]. Therefore, an Instance of a Framework was submitted to SWC. Additionally, considering the use of FOAF, we assume that Semblog is a Semantic Collaborative Tool.

Semblog seems to use a predefined ontology, called Personal Ontology for all of its components. This ontology gathers information about people (FOAF), interests of people (RDF Schema) and entries of a Weblog (RSS).

Semblog offers the following functionalities:

- Ontology Instances Editor functionality;
- Ontology Repository functionality;
- Semantic Recommender Policy functionality through RNA Alliance and FOAF Track-Back.

5.7 CS AKTive Space

CS AKTive Space (CAS) is a Semantic Web application that provides an integrated information overview of university-based Computer Science researchers, their works and their localization in the United Kingdom (UK). When developing the application, the authors had to face quite a few pragmatic challenges and decisions related to the Semantic Web: acquiring content; developing ontologies to mediate heterogeneous data sources; developing scalable RDF storage and query facilities; semantically directing interaction design; and facilitating knowledge-processing services over the harvested content [Shadbolt *et al.*, 2004].

Figure 16 presents the CAS system's core as a collection of Web services that communicate via HyperText Transfer Protocol (HTTP) and collaborate to provide the knowledge capabilities that the user interface requires.

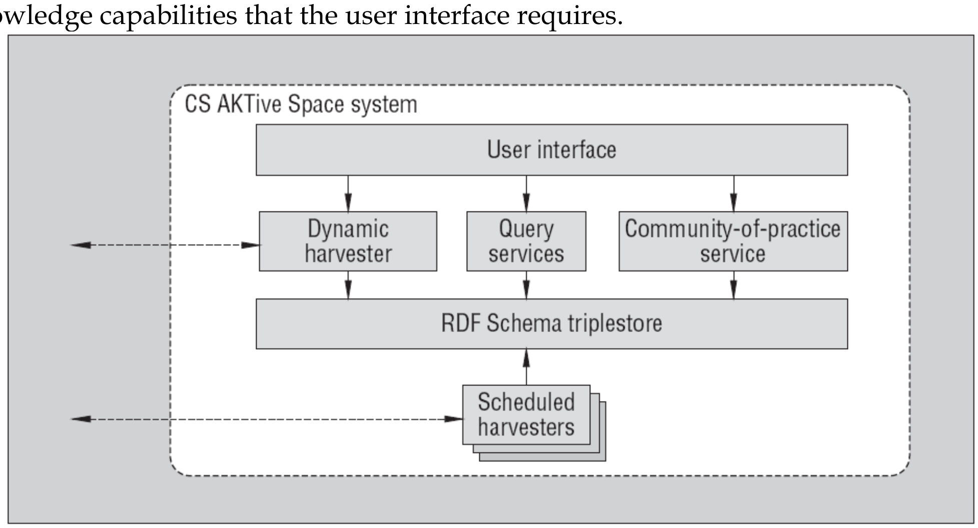

Figure 16 - Component interactions in the CAS system [Shadbolt *et al.*, 2004].

CAS has five main service types [Shadbolt *et al.*, 2004]:

- Scheduled harvesters: the harvesters extract information from Web sites, databases, spreadsheets, and other information sources, convert it into RDF using an appropriate ontology, and assert it into the triplestore;
- 3store: is the RDF Schema triplestore (knowledge base), which evaluates queries and performs simple inferences on the information the system uses;
- Dynamic harvester: this service takes instances that are underpopulated in the knowledge base (3store) and produces more knowledge about them;
- Community-of-practice service: is "Ontocopi". It uses ontological network analysis to discover connections between the objects that the ontology only implicitly represents;
- Geographic visualizer: this service provides a graphical representation of the geospatial information in the ontology (the locations of institutions of interest) and lets the user directly specify geographical constraints.

The query services type is not described as one of the five main service types, however it provides a kind of query preview [Plaisant *et al.*, 1999].
CAS is a Portal and can also be considered a Semantic Collaborative Tool given its use of a Community-of-practice service.

CAS uses a Wrappers and Mediators Integration functionality since it defines a single ontology to the system that has to be followed by the harvesters to translate information from other sources.

CAS offers the following functionalities:

- Browse functionality;
- Search functionality;
- Semantic Search functionality;
- Semantic Growth functionality through the dynamic harvesters;
- Multimedia Generation functionality through the geographic visualizer;
- Multimedia Metadata functionality through the permission to the user to directly specify geographical constraints; and
- Support for Diverse Languages functionality.

5.8 Semantic Web for Earth and Environmental Terminology (SWEET)

To support potential Semantic Web activities, a collection of ontologies for the Earth and environmental sciences and supporting areas were written by the Semantic Web for Earth and Environmental Terminology (SWEET) developers. SWEET is one of them [Raskin & Pan, 2003].

The developers used those ontologies in a prototype search tool that improves performance by creating additional relevant search terms based on the underlying semantics. They demonstrate how such a knowledge base can be "virtual" by adding a wrapper around remote, dynamic data repositories. The search tool consults the SWEET ontology to find related terms. These terms may be synonymous (same as), more specific (child of), or less specific (parent of) than those requested. The tool then submits the union of these terms to the Global Change Master Directory (GCMD) search tool and presents the results [Raskin].

The SWEET project and the SWEET description submitted to the SWC are somewhat contradictory. The products[43] of the SWEET project are a set of ontologies and an ontology-aided search tool. However, the description submitted to the SWC is quite different from that and we could not find a publication that reflected the submission. Therefore, we preferred not to consider this application on the SWC applications domain analysis.

5.9 BioInformatics

In the description of the BioInformatics submission to the SWC, the developers claim that the project applies Semantic Web technologies to integrate eight Web-based biological information sources for a sequence analysis service and search. A Web wrapper agent wraps the information sources as Web services. An ontology of agents is built so that it can represent the query answering power of each agent by specifying their input and output in RDF. However, we could not find a publication about the submission. Therefore, we preferred not to consider this application on the SWC applications domain analysis.

5.10 GeoShare

GeoShare is a cooperative project that intends to help the user: being she a professional of spatially referenced data, who needs to know which server contains the data and in which format; or a nonprofessional, such as a tourist, who would prefer digital maps presenting the requested information.

The GeoShare Network (Figure 17) employs a set of distributed, Web-based geoservices [Hübner *et al.*, 2004].

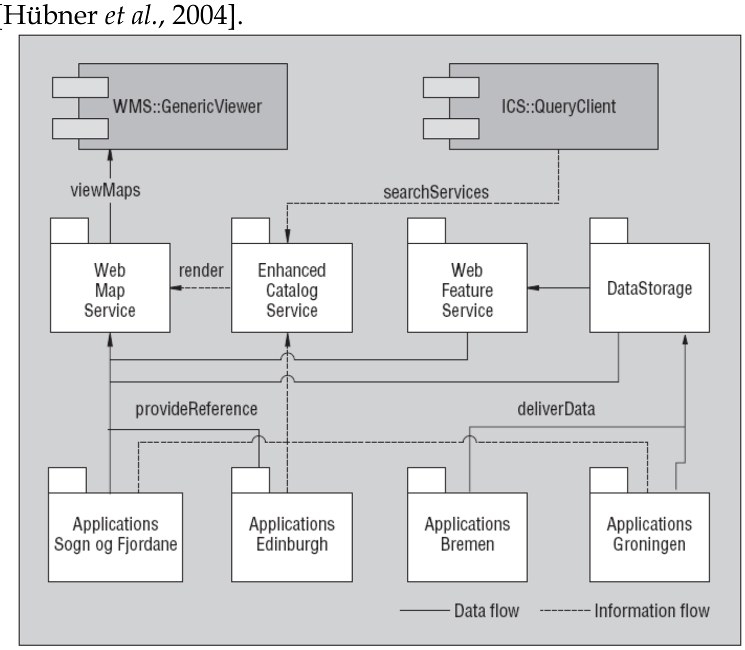

Figure 17 - The GeoShare Network [Hübner *et al.*, 2004].

43 SWEET Products - http://sweet.jpl.nasa.gov/products.html – accessed: 21/11/2006

Figure 17 presents a group of basic services that forms the backbone of the GeoShare Network. This service group consists of [Hübner *et al.*, 2004]:

- GeoShare DataStorage: A storage service that lets the project partners store geodata in several databases and data formats;
- GeoShare Web Feature Service: A service to provide full access to geodata stored in the GeoShare DataStorage or other data stores;
- GeoShare Enhanced Catalog Service: An online data catalog that provides search functionality and facilitates access to the Network's applications, services, and data; and
- GeoShare Generic Viewer (in combination with cascading Web Map Service): A tool to visualize digital maps individually or in an integrated, layered view.

Figure 18 depicts the GeoShare Enhanced Catalog Service. This is service is a central component of the GeoShare Network in all implementation phases. Through the catalog service, users can search and access all information sources that are registered within the system.

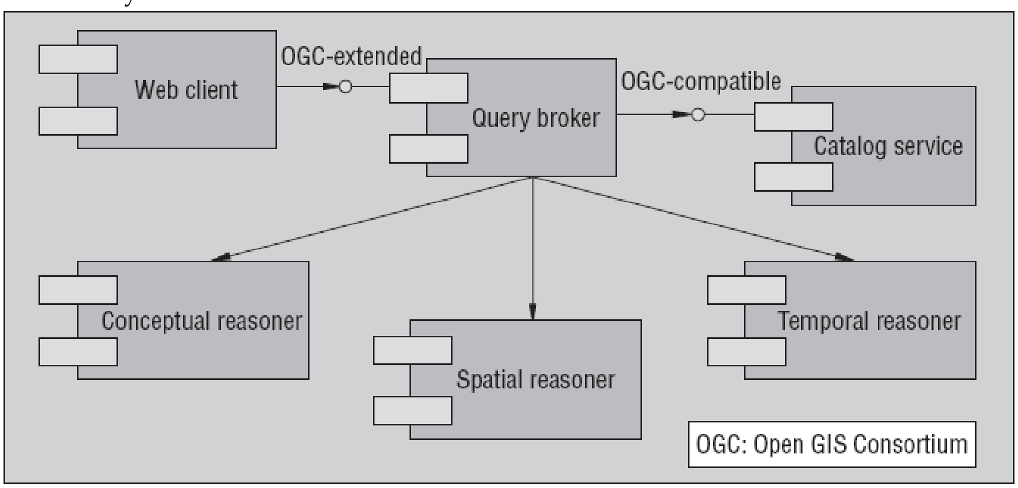

Figure 18 - The GeoShare Enhanced Catalog Service [Hübner *et al.*, 2004].

The GeoShare Enhanced Catalog Service goal is to be able to resolve complex information requests. The search module supports the specification of queries of the type *concept @ location* ***in*** *time.* That would explain the reasoners presented in Figure 18. Therefore, the service integrates two components: a standard Open Geospatial Consortium (OGC)-compliant catalog service and the Bremen University Semantic Translator for Enhanced Retrieval (BUSTER) as a tool to specify complex, knowledge-based queries [Hübner *et al.*, 2004].

From the user point of view, GeoShare offers data in two levels of abstractions: for professional users of spatially referenced data and for nonprofessional users in the form of digital maps. Therefore, we can consider GeoShare as a Portal.

GeoShare uses a kind of Wrappers and Mediators Integration functionality through the implementation of a set of distributed, Web-based geoservices.

GeoShare offers the following functionalities:

- Browse functionality;
- Search functionality;
- Semantic Search functionality;
- Semantic Query Expansion functionality;
- Multimedia Generation functionality; and
- Multimedia Metadata functionality.

5.11 SWC 2003 Summary

In Table 7, we summarize the applications submitted to the 2003's challenge; the functionalities they offer; their types and the type of integration they use. In the previous sections, we presented a brief explanation about each SWAPp as well as the functionalities they offer; their types and the type of integration they use.

Table 7 - SWC 2003 Summary

Applications		
1	SEmantic portAL (SEAL)	**Functionality** • Browse Functionality • Generation of Navigational Views Functionality • Search Functionality • Semantic Search Functionality • Support for Diverse Languages Functionality **Type of Application** • Portal • Instance of a Framework **Type of Integration** • Wrappers and Mediators Integration Functionality

Applications		
2	Drug Ontology Project for Elsevier (DOPE**)**	**Functionality** • Browse Functionality • Search Functionality • Semantic Search Functionality • Semantic Query Expansion Functionality • Multimedia Generation Functionality **Type of Application** • Portal • Instance of a Framework **Type of Integration** • Wrappers and Mediators Integration Functionality
3	SEmantic COllaboration (SECO**)**	**Functionality** • Browse Functionality • Search Functionality **Type of Application** • Portal **Type of Integration** • Wrappers and Mediators Integration Functionality
4	Annotated Terrestrial Information (AnnoTerra**)**	**Functionality** • Dynamic and Semantic Linking Hypertext Structures Functionality • Search Functionality • Semantic Search Functionality **Type of Application** • **Type of Integration** • Wrappers and Mediators Integration Functionality

Applications		
5	Building Finder	**Functionality** • Browse Functionality • Search Functionality • Semantic Search Functionality • Multimedia Metadata Functionality **Type of Application** • Portal **Type of Integration** • Wrappers and Mediators Integration Functionality
6	Semblog	**Functionality** • Semantic Recommender Policy Functionality • Ontology Instances Editor Functionality • Ontology Repository Functionality **Type of Application** • Instance of a Framework • Semantic Collaborative Tool **Type of Integration** • Manual Integration Functionality

Applications		
7	CS AKTive Space	**Functionality** • Browse Functionality • Search Functionality • Semantic Search Functionality • Support for Diverse Languages Functionality • Multimedia Metadata Functionality • Multimedia Generation Functionality • Semantic Growth Functionality **Type of Application** • Portal • Semantic Collaborative Tool **Type of Integration** • Wrappers and Mediators Integration Functionality
8	Semantic Web for Earth and Environmental Terminology (SWEET)	**Not considered**
9	BioInformatics	**Not considered**

Applications		
10	GeoShare	**Functionality** • Browse Functionality • Search Functionality • Semantic Search Functionality • Semantic Query Expansion Functionality • Multimedia Metadata Functionality • Multimedia Generation Functionality **Type of Application** • Portal **Type of Integration** • Wrappers and Mediators Integration Functionality

6 Semantic Web Challenge 2004 Applications

SWC 2003 applications used simple and shallow ontologies, and then the organizers set up the additional goal to SWC 2004 based on that observation. The additional goal was to show the benefits of the inference capabilities of the Semantic Web languages used by the applications. SWC 2004 had 18 submissions presented in the next sections and summarized in Section 6.19 (

Table 8).
The organizers of the challenge saw an increase in the use of reasoning in applications. However, most of the applications did not take benefit from inference capabilities, such as automatic classification or satisfiability checking [Klein & Visser, 2005].

6.1 DBin

DBin is a platform to build "Semantic Web Peer-to-Peer (P2P) communities". DBin establishes a use case where users can benefit from an assortment of semantic based activities such as browsing or intelligent interaction with the local media and files. DBin is composed of a number of experimental units to deal with specific kind of metadata (audio metadata extraction, textual analysis, desktop integration) as well as a domain oriented user interface. DBin also enables personalized trust policies to provide disregarding unwanted information [Tummarello *et al.*, 2005].

Figure 19 depicts the DBin architecture [Tummarello *et al.*, 2005]:

- At the database level, all the information is stored as RDF;
- Also contributing to the local database (DB) is a set of modules interacting with local and remote resources;
- The RDFGrowth algorithm (Growth Agent) is able to collect RDF metadata from other peers with common interests;
- DBin domain specific applications are called "Brainlets": Brainlets can be thought of "configuration packages" preparing DBin to operate on a specific domain (wine lovers, Italian opera fans etc.). Given that brainlets include customized user interface, the user might perceive brainlets as full "domain applications" which are run by DBin;

- The RDF DB undergoes a local trust based filtering and the resulting content, along with the data retrieved by the URI Bridge, explained later, is displayed by brainlets.

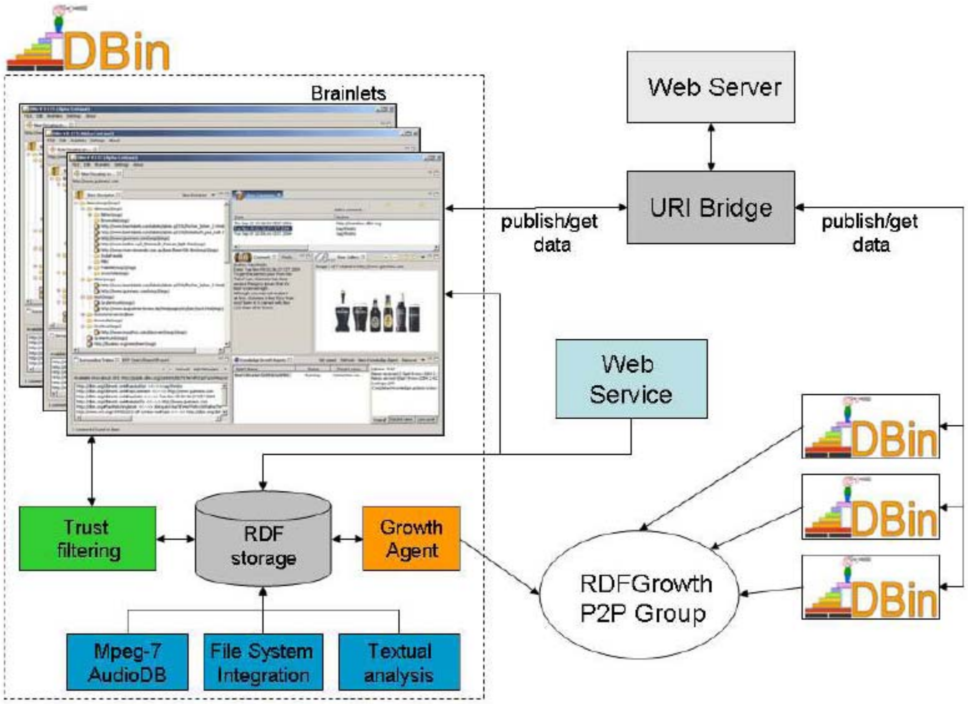

Figure 19 - A schema illustrating the overall DBin architecture and the use scenario [Tummarello *et al.*, 2005].

Also, as part of a "P2P community of DBin clients", there are some other units [Tummarello *et al.*, 2005]:

- The RDFGrowth P2P Group algorithm only exchanges pieces of RDF graphs. Therefore, some facility is needed to provide the user with actual content (images, text etc);
- Once a Uniform Resource Locator (URL) is available for a specific annotation, it is retrieved over standard HTML by the URIBridge upload/download facility;
- DBin clients exchange metadata through Growth Agents. Each DBin client, when publishing metadata referring to actual data, also makes sure this data is accessibly by publishing, if needed, in a Web space (Web Server).

As the SWC's organizers recognized, for some applications in the 2004 edition [Klein & Visser, 2005], DBin is an "infrastructure application" because it does not provide functionalities to the end-user[44]. Therefore, we preferred not to consider this application on the SWC applications domain analysis. On the other hand, the organizers also acknowledge that this kind of application is of extreme importance to the Semantic Web to become more widespread.

6.2 MusiDB

MusiDB is a partial implementation of a semantic portal that combines access to multiple sources with the use of recommendation techniques. The developers focus on the use of unique representations of data objects in public repositories (in this case MusicBrainz [Swartz, 2002]) and the use of recommendation mechanisms as a basis for supporting information access [Stegers *et al.*, 2006].

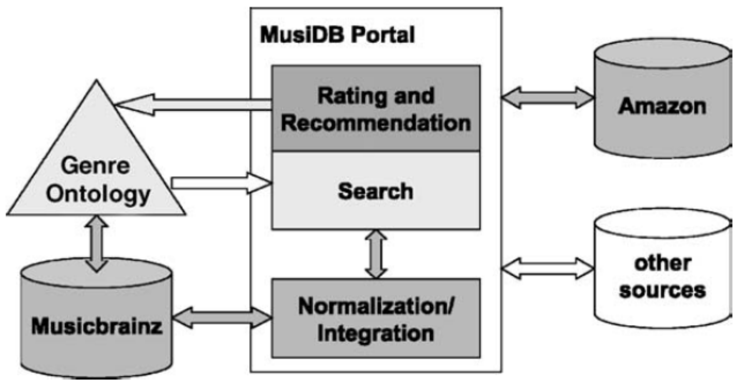

Figure 20 - Architecture of the MusiDB System [Stegers *et al.*, 2006].

Figure 20 depicts the architecture of MusiDB. MusiDB uses the MusicBrainz RDF database [Swartz, 2002]. Therefore, the search and recommendation functionality of the system uses the information from MusicBrainz as the primary representation to find relations between artists, albums and songs to expand incomplete user queries.

The system then links content from different sources to the instances returned by MusicBrainz. In the current implementation, the system links the Amazon Web services[45] with MusicBrainz to provide a list of available albums, their content and price.

In an experimental addition to the recommender system, the developers implemented a functionality that automatically assigns artists and albums from MusicBrainz to an ontology of musical genres based on user ratings. This functionality has the potential to be used for topic based search and recommendation [Stegers *et al.*, 2006].

MusiDB is Portal.

[44] Although the 2006 edition of the SWC is out of the scope of this work, DBin was resubmitted and won the third prize in that edition. In 2006, DBin is described under a new perspective and offers new functionalities.

[45] Amazon Web Services - http://aws.amazon.com - accessed: 16/06/2006.

MusiDB uses what could be considered a Wrappers and Mediators Integration functionality. MusiDB uses a simple data structure, which is used to uniformly represent the data from the data sources. All data is collected and combined at real-time based on the user query.

MusiDB offers the following functionalities:

- Browse functionality;
- Search functionality;
- Semantic Query Expansion functionality; and
- Semantic Recommender Policy functionality.

6.3 The Multilingual Access to Data Infrastructures of the European Research Area (MADIERA) Portal

The intention of the Multilingual Access to Data Infrastructures of the European Research Area (MADIERA) project[46] is to develop an infrastructure for the European social science community by integrating data with other tools, resources and products of the research process. The MADIERA portal is based on three main components [Alvheim & Ryssevik, 2005]:

- A common standard for data documentation developed by an international committee: Data Documentation Initiative[47] (DDI);
- The Multilingual European Language Social Science Thesaurus[48] (ELSST) that was used in the implementation of the DDI covering core concepts in social science research and methodology for nine European languages: English, French, Spanish, German, Greek, Norwegian, Danish, Finnish and Swedish;
- The Networked Social Science Tools And Resources[49] (NESSTAR) technology for making data resources available on the Web: NESSTAR is a state-of-the-art set of software tools developed to run data services at data archives.

The MADIERA Portal developed operates as a Web search engine by browsing and querying the NESSTAR Data Servers to harvest the RDF descriptions of the available statistical objects. The functionality of NESSTAR at the project initiation faced four basic aspects of the research process: resource location, metadata browsing, on-line analysis and data download.

To find and access appropriate resources, MADIERA's use of DDI and ELSST offers four different perspectives [Alvheim & Ryssevik, 2005]:

- Standard keyword and free-text searching (Google™ style);
- Browsing of structured subject-oriented catalogues (Yahoo® style);
- Geographical/map-based resource location (future work);

---

[46] MADIERA - http://www.madiera.net - accessed: 16/06/2006

[47] DDI - http://www.icpsr.umich.edu/DDI/index.html - accessed: 16/06/2006

[48] ELSST - http://www.limber.rl.ac.uk/Internal/Deliverables/D4_2_final_V2.doc - accessed: 16/06/2006

[49] NESSTAR - http://www.nesstar.com/ - accessed: 16/06/2006

- Specialized search for comparative data: this feature will establish "comparability" by analyzing a range of metadata descriptors.

The MADIERA portal, as its names indicates, is a Portal.

The MADIERA portal uses Wrappers and Mediators Integration functionality.

The MADIERA portal offers the following functionalities:

- Browse functionality;
- Search functionality;
- Semantic Search functionality;
- Support for Diverse Languages functionality;

6.4 SWAP

The Semantic Web Accessibility Platform (SWAP) is a knowledge-based approach to Web content accessibility. SWAP creates alternative representations (renderings) of sites, or SWAPviews, that enable people with varied special needs to access the content.

SWAP uses annotations, which reflect extra accessibility-related information about each page. A proxy server interprets these annotations to create an enhanced user experience including basic accessibility features required by users across platforms, and by international guidelines (Web Content Accessibility Guidelines[50] - WCAG) [Seeman, 2004].

We found only one publication about SWAP [Seeman, 2004]. However, by this publication we could not completely evaluate SWAP's functionality or implementation details. Therefore, we preferred not to consider this application on the SWC applications domain analysis.

6.5 SemanticOrganizer

SemanticOrganizer is a collaborative knowledge management application designed to support distributed project teams of NASA scientists and engineers. Although there are several document management tools available on the market, NASA science and engineering teams have some specialized requirements that justify solutions that are more specialized. For a list of them, please refer to the work of Keller, Berrios *et alli* [Keller *et al.*, 2004].

Some challenges were imposed by those requirements [Keller *et al.*, 2004]:

- To make the information easily and intuitively accessible to members of different collaborating teams;
- To develop a single application that could be rapidly customized to meet the needs of several different types of teams;
- To acquire knowledge and to do automatic ingestion of information;
- To provide rapid and precise access to repository information despite the large volume of data.

[50] WCAG - http://www.w3.org/WAI/intro/wcag.php - accessed: 16/06/2006

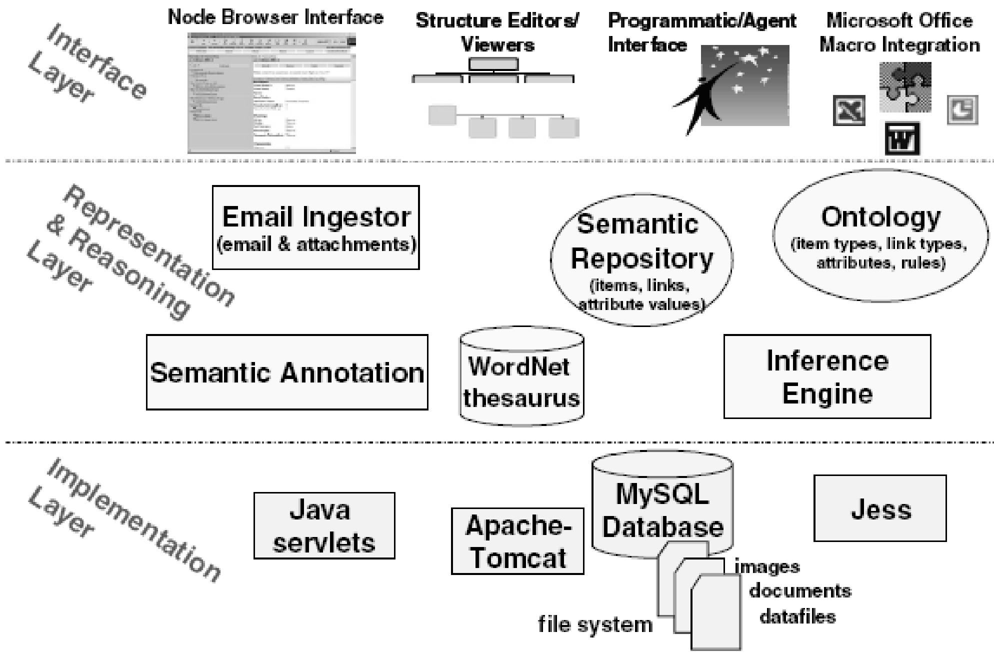

Figure 21 - SemanticOrganizer's architectural components [Keller *et al.*, 2004].

Figure 21 depicts SemanticOrganizer's components. For conceptual clarity, in the diagram the authors differentiate between the ontology and the semantic repository. However, they implement these components using a single representational mechanism that stores both classes and instances. Even though the repository is stored on a single server, access control and ontology customization mechanisms make the repository format and content appear different for each group of users. In essence, SemanticOrganizer is a set of virtual repositories, each built upon the same representational framework and storage mechanisms, but still customized to suit the needs of its specific users [Keller *et al.*, 2004].

SemanticOrganizer is a Portal. However, the authors differentiate their work from portals stating that (semantic) portals are intended to publish finalized work, while semantic repositories, as SemanticOrganizer, are intended to manage work products in various phases of a project lifecycle.

SemanticOrganizer uses only one ontology that can be customized by the teams. However, the authors state that the evolution is, and probably will still be, a difficult problem to address. The solution used now is cloning part(s) of the central ontology. This clone is evolved by the team that requested it instead of waiting for consensus about the changes that have to be made.

SemanticOrganizer offers the following functionalities:

- Browse functionality;
- Search functionality;
- Semantic Growth functionality;
- Multimedia Handling functionality;
- Ontology Instances Editor functionality; and

- Ontology Repository functionality.

6.6 Platypus Wiki

Platypus Wiki is a Personal Knowledge Management system, as well as a tool to manage Communities of Practice. Platypus Wiki represent metadata and relations between Wiki [Leuf & Cunningham, 2001] pages. It is a project enabling the collaborative editing of vocabularies and ontologies according to RDF Schema and OWL. The developers of Platypus Wiki decided to represent every RDF resource in the same way as a Wiki page. While standard Wikis use HTML links, Platypus Wiki uses RDF properties between resources to construct "labeled HTML links".

The convention chosen to represent a link to a page is namespace:pagename that can be reached with URL http://hostname/namespace/pagename/. If the user requests a URL http://hostname/namespace/pagename/index.rdf, the system returns only RDF metadata about the resources. Similarly if the user asks for http://hostname/namespace/pagename/index.html, the choice was to return only the plain HTML content without any navigation bar, page header or footer. [Tazzoli *et al.*, 2004].

Platypus Wiki is a Semantic Wiki. It can also be considered a Portal and an Ontology Tool.

Platypus Wiki uses one ontology at a time.

Platypus Wiki offers the following functionalities:

- Ontology Schema Editor functionality;
- Ontology Instances Editor functionality;
- Ontology Repository functionality;
- Browse functionality;
- Search functionality; and
- Semantic Search functionality.

6.7 MuseumFinland

MuseumFinland - Finnish Museums on the Semantic Web[51] is a system that presents an inter-museum exhibition of over 4000 cultural artifacts. MuseumFinland system also incorporates metadata concerning some 260 historical sites in Finland. The goals for developing the system were [Hyvönen *et al.*, 2005]:

- Global view to distributed collections;
- Content-based information retrieval;
- Semantically linked contents;
- Easy local content publication.

---

[51] MuseumFinland - http://museosuomi.fi - accessed: 12/06/2006 on Google's cache.

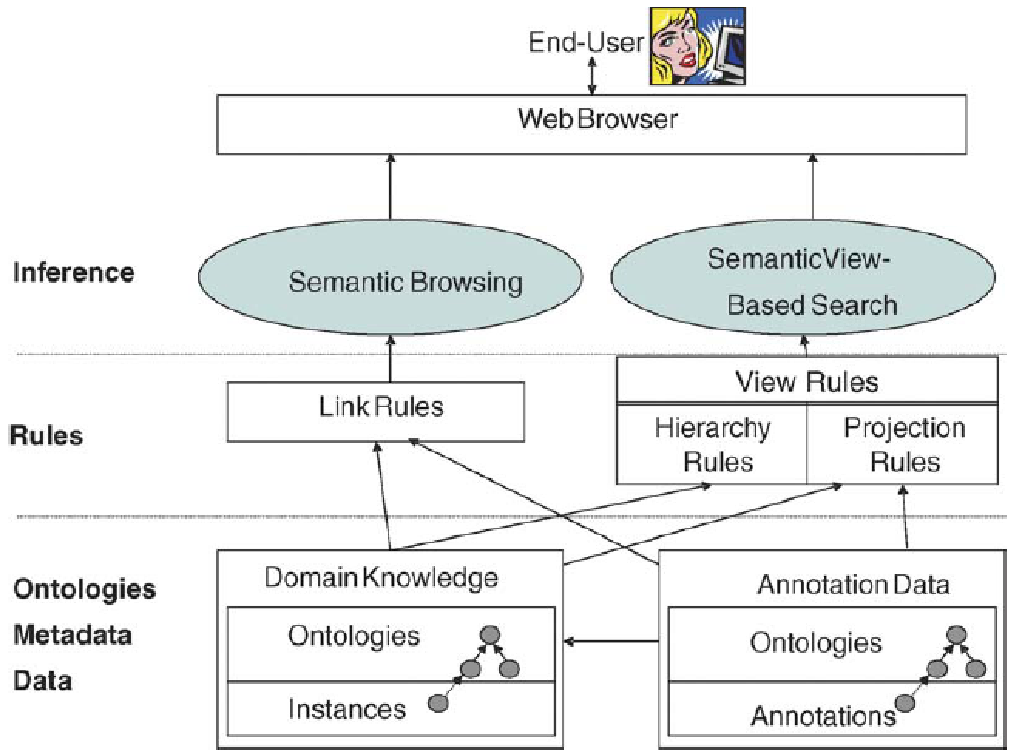

Figure 22 - Architecture of MuseumFinland on the server side [Hyvönen *et al.*, 2005].

Figure 22 illustrates the architecture of MuseumFinland. The architecture separates generic search and browsing services from the underlying application dependent schemas and metadata by a layer of logical rules. According to this separation, the portal creation framework and software developed could be of use in other domains too.

MuseumFinland has been implemented by using a tool called OntoViews[52] [Mäkelä *et al.*, 2004]. OntoViews consists of the three major components presented in Figure 23.

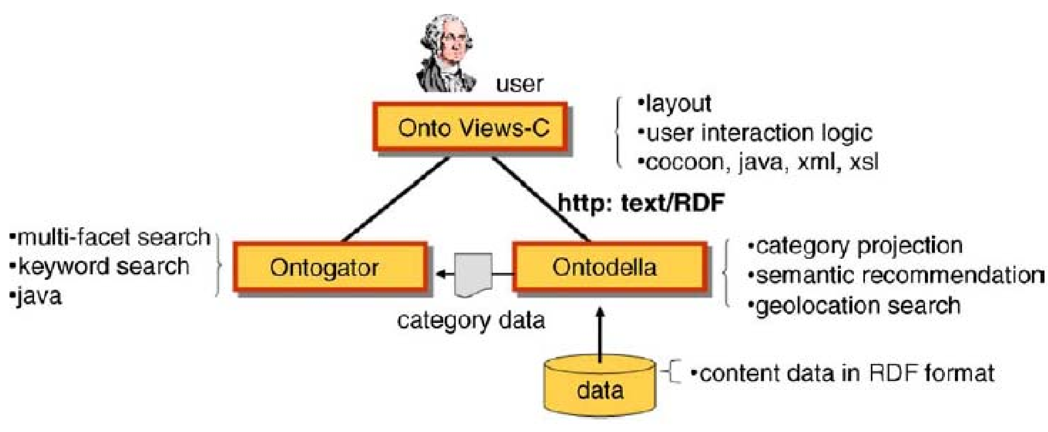

Figure 23 - The components of OntoViews [Hyvönen *et al.*, 2005].

---

52 http://www.seco.tkk.fi/projects/semweb/dist.php - accessed: 22/09/2006

OntoViews-C component merges the services of Ontogator and Ontodella together, and provides the user interfaces. The logic server Ontodella provides the system with reasoning services, such as category view projection and dynamic semantic link recommendations; The search engine Ontogator is a generic view-based RDF search engine, responsible for the multi-facet search functionality of the system [Hyvönen *et al.*, 2005].

MuseumFinland is a Portal and an Instance of a Framework (OntoViews-C). MuseumFinland uses a consolidated global repository. However, for each collection a content creation process is carried out. We consider, then, that MuseumFinland uses Wrappers and Mediators Integration functionality.

MuseumFinland offers the following functionalities:

- Browse functionality;
- Generation of Navigational Views functionality;
- Search functionality;
- Semantic Search functionality;
- Multimedia Metadata functionality;
- Access through Diverse Devices functionality; and
- Semantic Recommender Policy functionality.

6.8 Knowledge Management Platform (KmP)

The objective of the Knowledge Management Platform (KmP) project is to increase the collection of competences of the Telecom Valley of Sophia Antipolis - France by supporting actors in stating their interests and needs in a shared space. The solution relies on the specification, design, building and evaluation of an online customizable Semantic Web application [INRIA, 2006].

This Web application relies on ontology-based models and inferences; and merges the frameworks of the Semantic Web (RDF, RDF Schema), the classic Web (HTML, Cascading Style Sheets - CSS, Scalable Vector Graphics - SVG) and the structured Web (XML, eXtensible Stylesheet Language Transformation -XSLT) to integrate data coming from very different sources. The application allows queries from different viewpoints, adapt content to users, analyze, group, infer and render indicators of the Telecom Valley situation [INRIA, 2006].

KmP relies on the integration of multiple components: databases for back-end persistence, Web servers with Java Server Pages (JSP) and servlets to provide front ends, and the CORESE Semantic Web server[53] to provide Semantic Web processing capabilities [INRIA, 2006].

At the time of this writing, we could not find a paper about KmP, the information we got is from the project homepage [INRIA, 2006]. However, by that information on the homepage we could not completely evaluate KmP's functionality or implementation details. Therefore, we preferred not to consider this application on the SWC applications domain analysis.

[53] CORESE - http://www-sop.inria.fr/acacia/soft/corese/ - accessed: 16/06/2006

6.9 pOWL

pOWL is intended to be a comprehensive ontology management tool. It integrates diverse aspects of ontology management such as storage and querying, supplying an API and a collaborative Web user interface.

pOWL's architecture consists of 4 stacked tiers. The architecture tries to minimize dependencies and to supply clean interfaces between tiers. The 4 tiers are [Auer, 2005]:

- pOWL store - Structured Query Language (SQL) compatible relational database back-end;
- RDFAPI, RDFSAPI, OWLAPI - layered APIs for handling RDF, RDF Schema and OWL;
- pOWL API - containing classes and functions to build Web applications on top of the previous APIs;
- User interface - a set of Hypertext Preprocessor (PHP) pages combining widgets provided by pOWL API for accessing (browsing, viewing, editing) model data in a pOWL store.

As the SWC's organizers recognized, for some applications in the 2004 edition [Klein & Visser, 2005], pOWL is an "infrastructure application" because it does not provide functionalities to the end-user. Therefore, we preferred not to consider this application on the SWC applications domain analysis. On the other hand, the organizers also acknowledge that this kind of application is of extreme importance to the Semantic Web to become more widespread.

6.10 Semantic Portal of International Affairs (SPIA)

The Semantic Portal of International Affairs (SPIA) provides semantic access (contrasting to "pure" keyword-based access) to content. In SPIA's case, a semantic access approach is provided through tools and techniques that are being developed in the context of several European and National (Spanish) research and development projects [Contreras *et al.*, 2004]:

- Semantic search engine;
- Semantic publishing and navigation;
- Three-dimensional (3D) Visualization.

To implement these features of semantic access, the components of SPIA include:

- An ontology in the domain of International Affairs;
- An automatic annotator (metadata generator), named Knowledge Parser® (Figure 24);
- A semantic search engine with a natural language interface, as well as a forms based interface;
- A publication tool for publishing semantic content on the Web, called Duontology®, enabling semantic navigation including a 3D visualization tool.

Knowledge Parser® is able to parse content and extract knowledge from it. Figure 24 presents the process that is executed in three main steps: Source Preprocessing, Information Identification and Ontology Population. For an extensive explanation about those steps, please refer to work of Contreras, Benjamins *et alli* [Contreras *et al.*, 2004].

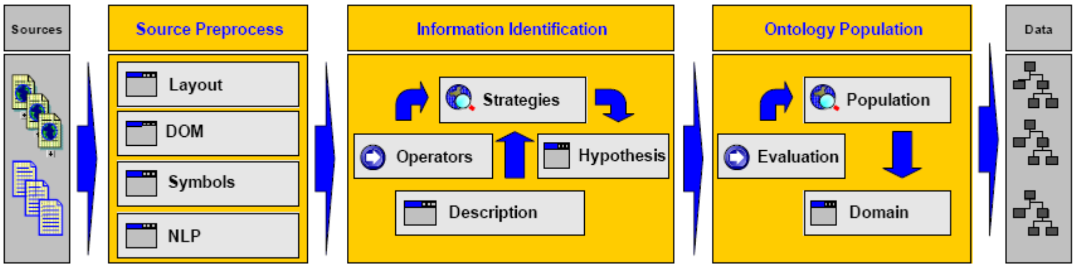

Figure 24 - Overview of the extraction and population process [Contreras *et al.*, 2004].

In SPIA, the Knowledge Parser® executes two roles. The first is the wrapping of the USA's Central Intelligence Agency (CIA) World Factbook Web in order to populate the ontology with instances with information regarding countries such as their government composition, geographical data, political and commercial agreements etc. Once the process populates the ontology with instances, the Knowledge Parser® executes its second role, which is to be applied to the documents provided by the Spain's Real Instituto Elcano de Estudios Internacionales y Estratégicos (Elcano Institute).

The Semantic Search Engine developed, with a natural language interface as well as a forms based interface, returns instances that constitute answers to queries instead of documents containing searched strings as traditional keyword based engines would do.

The developers of SPIA emphasize that the knowledge base as modeled by domain experts and knowledge engineers is not always a good candidate to visualize as is. Therefore, they introduce the idea of a "visualization ontology", which makes explicit all visualization rules and allows an uncomplicated interface management. This ontology contains concepts and instances (publication entities) as perceived on the interface by the end user, and the visualization ontology returns the attribute values from the International Relations ontology using a query. Consequently, not duplicating content [Contreras *et al.*, 2004].

SPIA is a Portal with advanced semantic functionalities.

SPIA uses a Wrappers and Mediators Integration functionality.

SPIA offers the following functionalities:

- Browse functionality;
- Generation of Navigational Views functionality;
- Search functionality;
- Semantic Search functionality;
- Multimedia Generation functionality;
- Support for Diverse Languages functionality; and
- Access through Diverse Devices functionality.

6.11 Unspecified Ontology (UNSO)

Unspecified Ontology (UNSO) approach supposes that the domain ontology is not fully defined and peers can dynamically specify parts of the ontology. UNSO approach recommends a more flexible manner to describe an object. It allows constructing a multi-layered hypercube (MLH) graph topology, supporting efficient semantic routing [Ben-Asher & Berkovsky, 2004].

To reach the semantic routing, HyperCup's [Schlosser *et al.*, 2002] hypercube graph topology was extended to a multi-layered hypercube (MLH). HyperCup proposes a scalable and efficient ontology-based graph topology to cluster peers in a P2P network. In HyperCup, a set of known ontologies is used to categorize peers as providers of particular services to efficiently route and broadcast queries [Schlosser *et al.*, 2002]. Using UNSO does not force peers to share or to use any explicit ontology [Ben-Asher & Berkovsky, 2004].

The work submitted to the challenge scrutinized the issue of implementing an infrastructure, dedicated for e-Commerce transactions over P2P networks. The service, provided by the system is insertion, searching and a matching of appropriate demand and supply ads (e-Commerce advertisements). Briefly, the main contribution of UNSO is the notion of ontologies (as a technique for managing a dynamic set of forms) and its accompanied semantic routing [Ben-Asher & Berkovsky, 2004].

UNSO is a Semantic P2P application and could also be considered an Ontology Tool.

UNSO uses an unspecified ontology, only a small part of it is fixed and Wordnet[54] is used to eliminate ambiguity and enhance system precision for the terms used by the peers.

UNSO offers the following functionalities:

- Search functionality;
- Semantic Query Expansion functionality;
- Ontology Schema Editor functionality;
- Ontology Instances Editor functionality; and
- Ontology Repository functionality;

6.12 Semantic Web Assistant

The Semantic Web Assistant is part of a thesis submitted to the Department of Computer Science at the University of Applied Sciences Bonn-Rhein-Sieg, Germany. The thesis explores the possibilities of a combination of Semantic Web technologies with production rule systems for letting end-users discover some of the applications of the Semantic Web.

However, we could not find a publication about the application. Therefore, we preferred not to consider this application on the SWC applications domain analysis.

[54] Wordnet - http://www.cogsci.princeton.edu/~wn – Accessed: 01/04/2006

6.13 Swoogle

Swoogle [Ding *et al.*, 2004] intends to help human users and software agents find pertinent knowledge on the Semantic Web. The Swoogle search engine discovers, indexes, and analyzes the ontologies and facts that are encoded in Semantic Web documents (SWD) [Li *et al.*, 2005].

Rather than using one regular crawling technique to discover SWDs, Swoogle employs a fourfold strategy [Li *et al.*, 2005]:

- running metasearches on conventional Web search engines, such as Google™, to find candidates;
- using a focused Web crawler to traverse directories in which SWDs have been found;
- harvesting URLs when processing discovered SWDs; and
- collecting URLs of SWDs and directories containing SWDs that users have submitted.

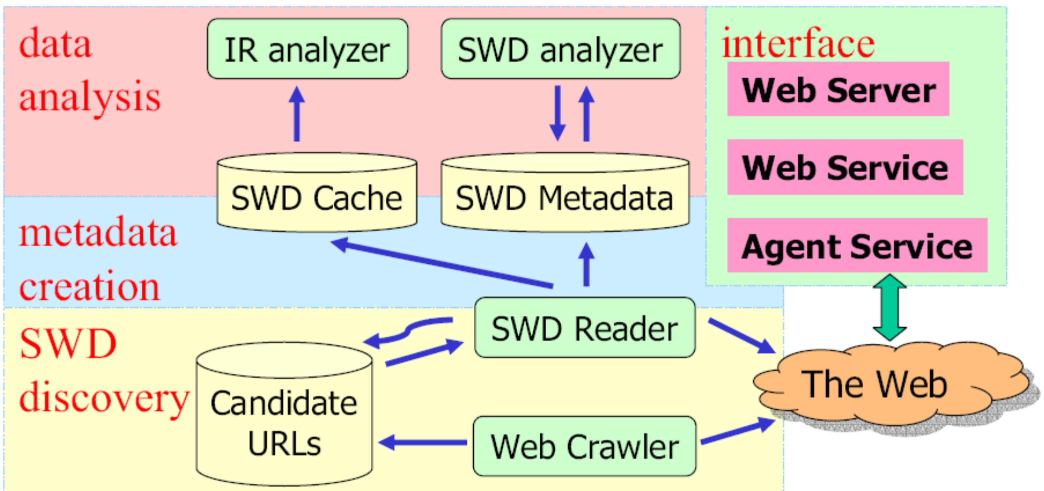

Figure 25 - The architecture of Swoogle [Ding *et al.*, 2004]

Figure 25 presents that Swoogle's architecture can be broken into four major components:

- SWD discovery;
- Metadata creation;
- Data analysis; and
- Interface.

This architecture is data centric and extensible, components work independently and interact with one another through a database [Ding *et al.*, 2004].

As the SWC's organizers recognized, for some applications in the 2004 edition [Klein & Visser, 2005], Swoogle is an "infrastructure application" because it does not provide functionalities to the end-user. Therefore, we preferred not to consider this application on the SWC applications domain analysis. On the other hand, the organizers also acknowledge that this kind of application is of extreme importance to the Semantic Web to become more widespread.

6.14 Flink

Flink is a presentation of the professional and social connectivity of Semantic Web Researchers. Flink has three objectives [Mika, 2005a]:

- To be a demonstration of the use of Semantic Web technology;
- To be a portal for any person who is interested to learn about the (work of the) Semantic Web community; and
- To have its collected data used for the purposes of social network analysis, in particular learning about the nature of power and innovativeness in scientific communities.

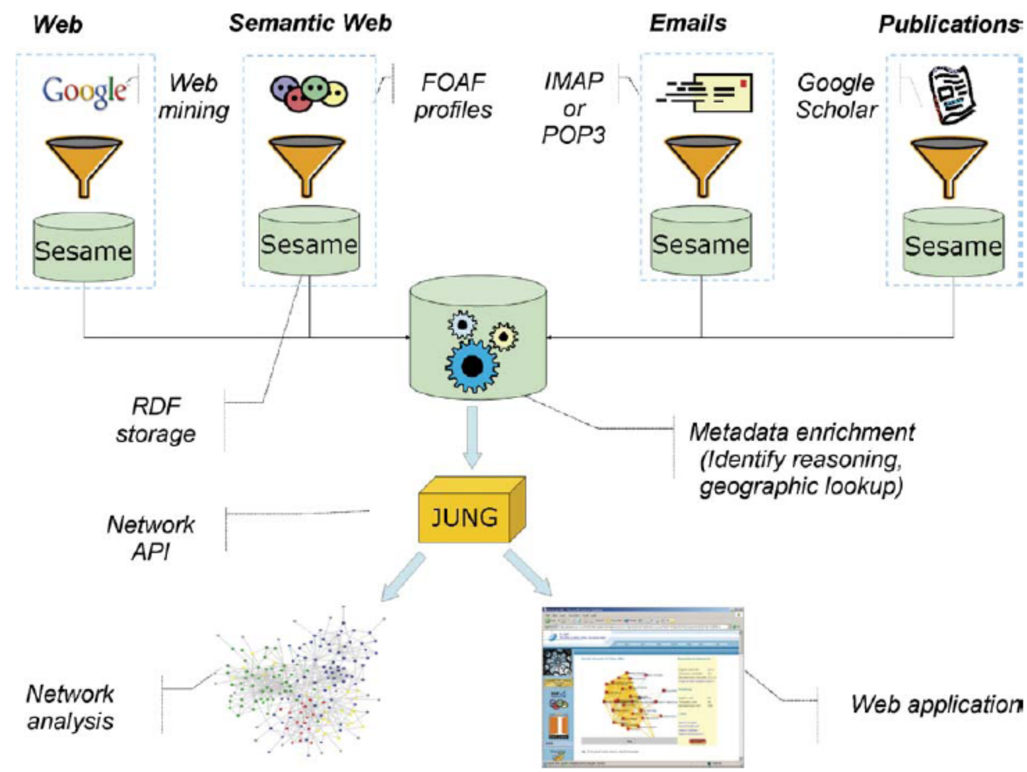

Figure 26 - The architecture of Flink from metadata acquisition (top) to the user interface (bottom) [Mika, 2005a].

The Flink's author suggests the segregation of the architecture of Flink in three layers related to metadata acquisition, storage (representation, inference and storage) and visualization as seen in Figure 26 from top to bottom.

Flink is a Portal, although it does not offer traditional search functionality. This is not a problem because the list of Semantic Web Researchers for the website was limited, due to practical implications at the time of this writing, to those who have been Chairs, Program Committee members or authors of full papers at any of the past international Semantic Web events (SWWS'01, ISWC2002, ISWC2003, ISWC2004 and ISWC2005). A deeper study of the Network Analysis area and its relationship (or similarity) with collaborative applications would be necessary to say if Flink is also a Semantic Collaborative tool.

Flink uses a Wrappers and Mediators Integration functionality.

Flink offers the following functionalities:

- Browse functionality; and
- Multimedia Generation functionality.

6.15 Bibster

Bibster is a P2P system for exchanging bibliographic data among researchers [Haase *et al.*, 2004]. Bibster is an instance of the Semantic Web and Peer-to-Peer Project - System Architecture (SWAPSA). For more information about SWAPSA, please refer to [Broekstra *et al.*, 2003] and [**SWAP EU IST-2001-34103 Final Report**, 2004].

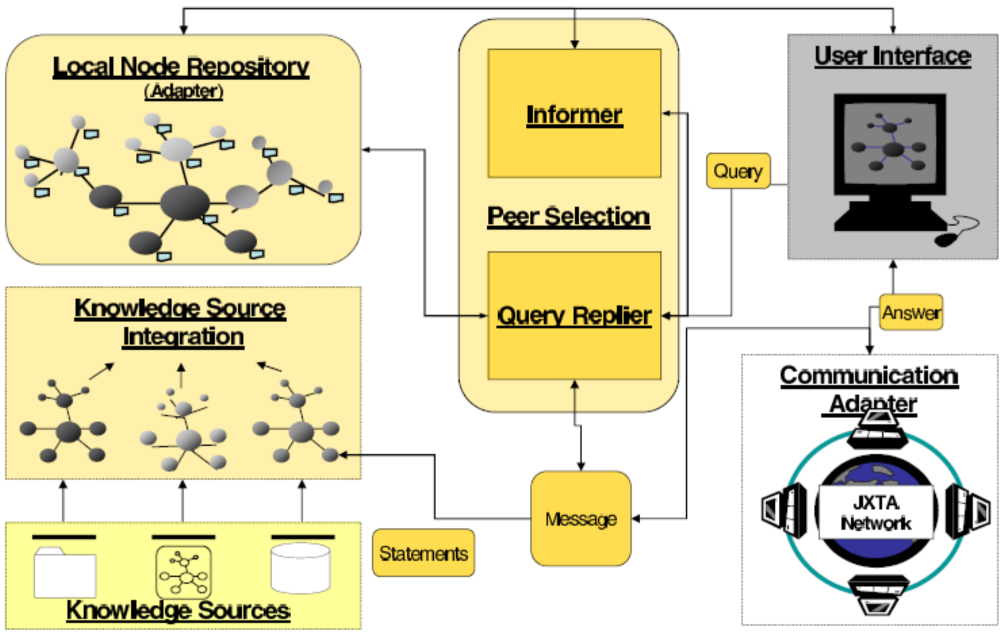

Figure 27 - SWAP System Architecture [Haase *et al.*, 2004].

Figure 27 shows an abstract design of the components of the architecture of a node in the P2P system. Next, the components are briefly described as instantiated for the Bibster system [Haase *et al.*, 2004]:

- Communication Adapter: It serves for sending and forwarding queries for the rest of the system. It encapsulates all low-level communication details from other parts of the system;
- Knowledge Sources;
- Knowledge Source Integrator: The Knowledge Source Integrator extracts and integrates internal and external knowledge sources into the Local Node Repository;
- Local Node Repository:
    - ♦ Mediates between views and stored information;
    - ♦ Supports query formulation and processing;
    - ♦ Specifies the peer's interface to the network;
    - ♦ Provides the basis for peer ranking and selection.

- Informer: It proactively advertises the available knowledge of a peer in the P2P network and it discovers peers with knowledge that may be relevant for answering the user's queries;
- Query Replier: It is the coordinating component controlling the process of distributing queries. It receives queries from the user interface or from other peers;
- User Interface: The user interface allows the user to import, create and edit bibliographic metadata as well as to easily formulate queries.

Bibster is Semantic P2P Application, a Portal and an Instance of a Framework. Bibster seems to uses only one integrated ontology that is defined by the domain of bibliographic metadata and ACM Topic Hierarchy. However, Bibster seems to use a kind of Wrappers and Mediators Integration functionality for integrating local knowledge sources.

Bibster offers the following functionalities:

- Browse functionality;
- Search functionality;
- Semantic Growth functionality through its semantic duplicate detection;
- Ontology Instances Editor functionality; and
- Ontology Repository functionality.

6.16 Mediator EnvirOnment for Multiple Information Sources (MOMIS)

Mediator EnvirOnment for Multiple Information Sources (MOMIS) is a framework that extracts and integrates information of heterogeneous sources [Beneventano & Bergamaschi, 2004]. Figure 28 presents the MOMIS architecture. The MOMIS framework is based on a language and two main components [Bergamaschi *et al.*, 2005] :

- The ODL-I3 language that extends an object-oriented language (Object Definition Language - ODL) with an underlying Description Logic. The language is derived from the standard ODL-ODMG [Cattell & Barry, 2000];

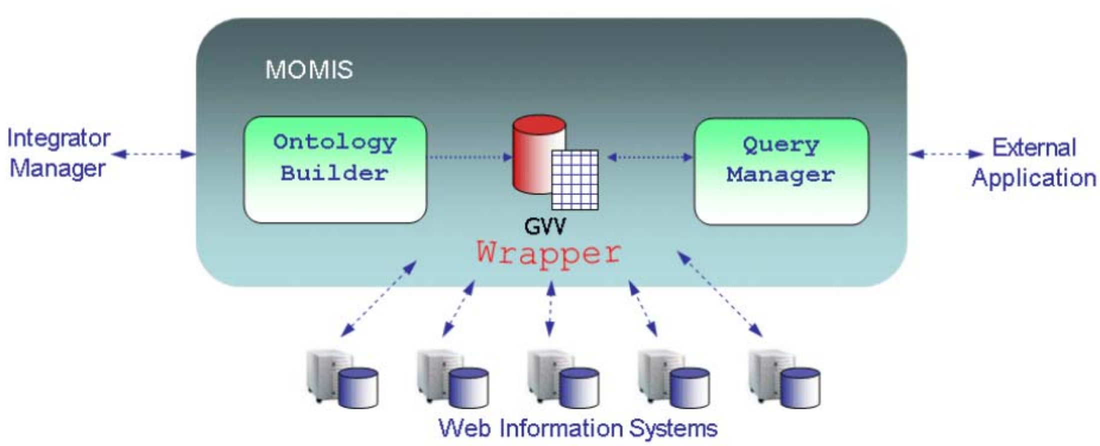

Figure 28 - The MOMIS Architecture [Bergamaschi *et al.*, 2005].

- The Ontology Builder; MOMIS system implements a semi-automatically data integration, developed in accordance with the Global as View (GAV) approach. The result of the integration process is a global schema, which provides a reconciled, integrated and virtual view of the underlying sources, Global Virtual View (GVV). The GVV is a collection of (global) classes that represent the information contained in the sources, and it is the result of the integration process. The GVV is then semi-automatically annotated according to a lexical ontology. The implementers of MOMIS firstly markup the local metadata descriptions and then the MOMIS system generates an annotated conceptualization of the sources. Their approach "constructs" the domain ontology as the synthesis of the integration process, despite the fact that the common approach in the Semantic Web is supported by "a priori" developed ontology [Beneventano & Bergamaschi, 2004]. The information integration process for building the GVV is shown in Figure 29;

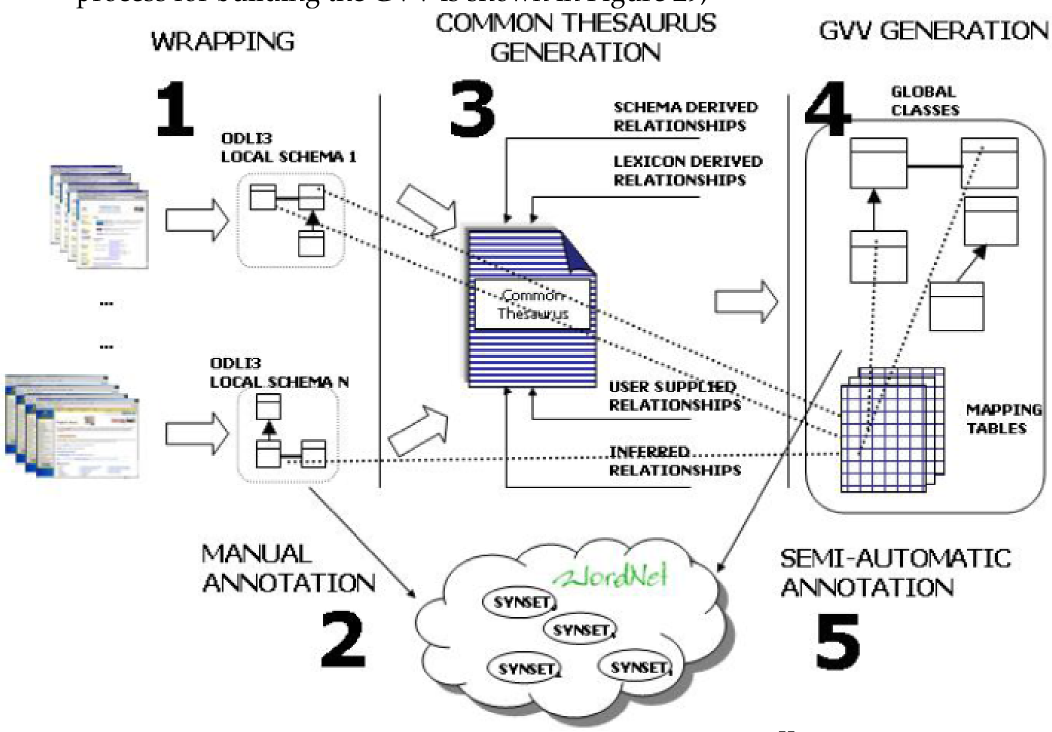

Figure 29 - Overview of the ontology-generation process...[55] [Beneventano & Bergamaschi, 2004]

- The MOMIS Query Manager (Figure 28) is a coordinated set of functions that takes a query, decomposes it according to the mapping of the GVV on the local data sources relevant to the query. Query Manager sends the subqueries to those data sources, collects their answers, performs any residual filtering necessary, and finally delivers the answer to the user [Bergamaschi *et al.*, 2005].

55 Continuation of the caption: "The figure shows the local schemas' generation, where local schemas are annotated according to the lexical ontology WordNet, the Common Thesaurus generation, and finally the GVV global classes. In particular, these ones are connected by means of mapping tables to the local schemas and are (semi-automatically) annotated according to WordNet." [Beneventano & Bergamaschi, 2004]

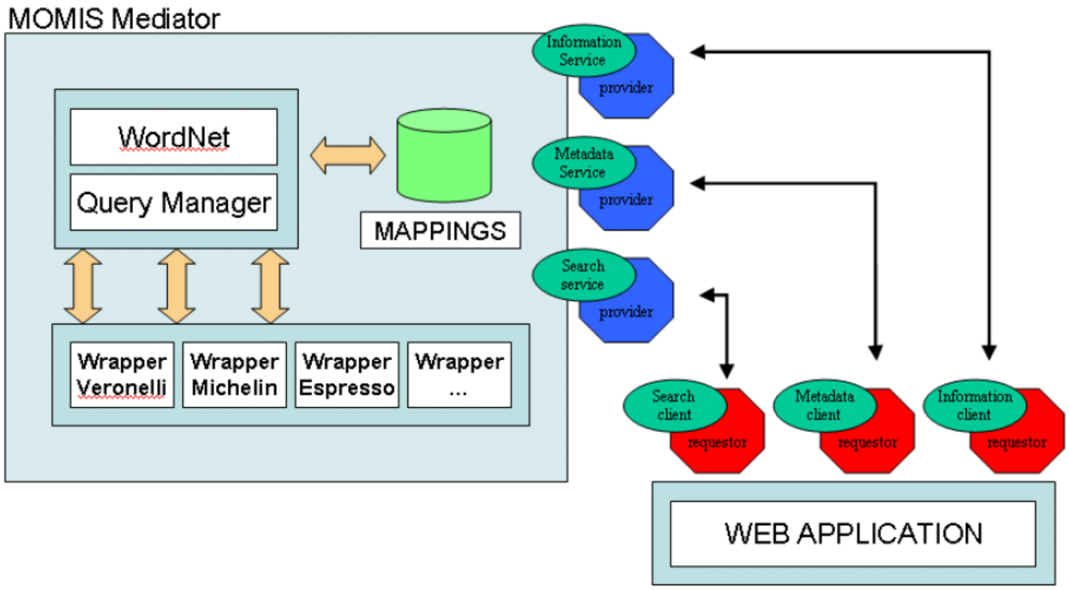

Figure 30 - The MOMIS Web services architecture [Bergamaschi *et al.*, 2005].

An instantiation of the MOMIS framework is a system (Figure 30) that is based on a conventional wrapper/mediator architecture, and provides methods and open tools for data management in Internet-based information systems [Bergamaschi *et al.*, 2005]. The instance submitted to the SWC was of the tourist domain based on the restaurants industry. In [Bergamaschi *et al.*, 2005] it is presented the new Web Services architecture for MOMIS instead of the CORBA-2 architecture used in [Beneventano & Bergamaschi, 2004].

The MOMIS instance submitted to SWC was a Portal and an Instance of a Framework.

MOMIS uses Wrappers and Mediators Integration functionality.

MOMIS offers the following functionalities:

- Browse functionality; and
- Search functionality.

6.17 Annotea Shared Bookmarks

Annotea [Kahan *et al.*, 2001] is a Semantic Web based project which observed what users did naturally and opted for common metaphors for supporting better collaboration [Koivunen, 2005].

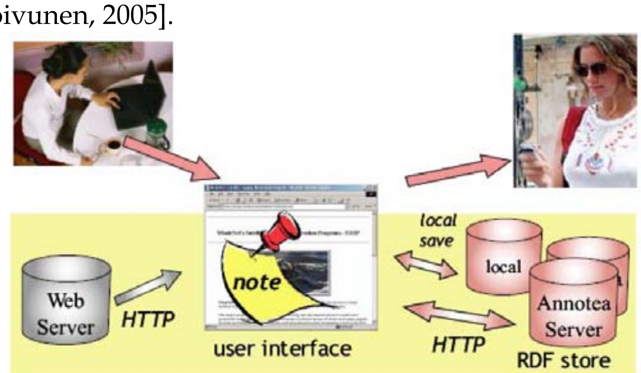

Figure 31 - The basic Annotea architecture [Koivunen, 2005].

Figure 31 presents the basic Annotea architecture. In the architecture, there are various RDF metadata repositories storing Annotea objects, a user interface providing different views to the objects in the context of the Web documents or other Web resources, and users collaborating via these objects [Koivunen, 2005].

The content of the Annotea objects can be viewed in any Web browser as XML text. However, to be usable for any user the normal Web browser needs to support Annotea metaphors. For example, in Mozilla/Firefox®, the tool Annotea Ubimarks[56] provides that functionality.

Annotea objects metadata can be stored locally, in Annotea servers or as published collections of Annotea objects in Web documents. The biggest direct benefit from the use of Semantic Web technologies and metadata in Annotea objects is that the user generated metadata can be easily combined and reused in several other applications, such as user profiles for services, data mining and search engine applications [Koivunen, 2005].

Annotea Shared Bookmarks is an Ontology Tool.

Annotea Shared Bookmarks seems to use only one ontology: the Annotea Metaphors ontology.

Annotea Shared Bookmarks offers the following functionalities:

- Ontology Instances Editor functionality;
- Ontology Repository functionality; and
- Access through Diverse Devices functionality.

6.18 GOHSE

GOHSE is an application of the Conceptual Open Hypermedia Service (COHSE) [Carr *et al.*, 2001] architecture to Bioinformatics, using the Gene Ontology (GO) [Ashburner *et al.*, 2000] as an ontology and GO associations as link targets. GOHSE provides both glossary functionality and the possibility of building dynamic hypertext structures linking bioinformatics documents [Bechhofer *et al.*, 2005].

The COHSE system enhances document resources through the dynamic addition of hypertext links. These links are derived using an ontology and associated lexicon along with a mapping from concepts to possible link targets.

The implementation of the system (GOHSE) is in the form of a COHSE agent (Figure 32), in conjunction with two services: the Ontology Service and the Annotation Service. The agent adds, to documents, links based on the semantic content of those documents. The Ontology Service sends ontological information in a dynamic fashion to the agent. The Annotation Service correlates concepts with resources and provides mechanisms for querying those associations. In the implementation, the agent is attached to a proxy through which all HTTP requests are routed [Bechhofer *et al.*, 2005].

[56] Annotea Ubimarks - http://www.annotea.org/mozilla/ubi.html - accessed: 16/06/2006.

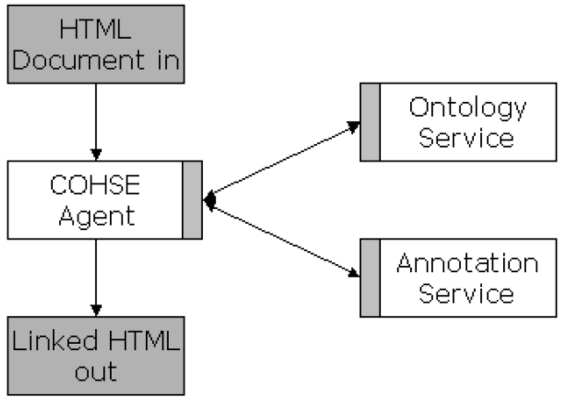

Figure 32 - COHSE Architecture [Bechhofer *et al.*, 2005].

COHSE extends the Distributed Link Service (DLS) [Carr *et al.*, 1995] with ontological services, providing information relating to an ontology. These services include mappings between concepts and lexical labels (synonyms). The services also provide information about relationships, such as sub- and super-classes.

DLS is an Open Hypermedia System [Grønbæk *et al.*, 1999] [Østerbye & Wiil, 1996] which rather than embedding links in the documents, consider them first class citizens. They are stored and managed separately from the documents and can thus be stored, transported, shared and searched separately from the document itself. Documents and linkbases are dynamically combined by the DLS, which then adds proper links to documents [Bechhofer *et al.*, 2005].

GOHSE is an Instance of a Framework (COHSE).

GOHSE uses Wrappers and Mediators Integration functionality.

GOHSE offers the following functionalities:

- Dynamic and Semantic Linking Hypertext Structures functionality;
- Semantic Query Expansion functionality.

6.19 SWC 2004 Summary

In

Table 8, we summarize the applications submitted to the 2004's challenge; the functionalities they offer; their types and the type of integration they use. In the previous sections, we presented a brief explanation about each SWAPp as well as their functionalities, their types and the type of integration they use.

Table 8 - SWC 2004 Summary

<table>
<tr><th colspan="2">Applications</th><th></th></tr>
<tr><td>1</td><td>DBin</td><td>Not considered</td></tr>
<tr><td>2</td><td>MusiDB</td><td>Functionality
Browse Functionality
Search Functionality
Semantic Query Expansion Functionality
Semantic Recommender Policy Functionality
Type of Application
Portal
Type of Integration
Wrappers and Mediators Integration Functionality</td></tr>
<tr><td>3</td><td>The Multilingual Access to Data Infrastructures of the European Research Area (MADIERA) Portal</td><td>Functionality
Browse Functionality
Search Functionality
Semantic Search Functionality
Support for Diverse Languages Functionality
Type of Application
Portal
Type of Integration
Wrappers and Mediators Integration Functionality</td></tr>
<tr><td>4</td><td>SWAP</td><td>Not considered</td></tr>
</table>

Applications		
5	SemanticOrganizer	**Functionality** • Browse Functionality • Search Functionality • Multimedia Handling Functionality • Semantic Growth Functionality • Ontology Instances Editor Functionality • Ontology Repository Functionality **Type of Application** • Portal **Type of Integration** • Manual Integration Functionality
6	Platypus Wiki	**Functionality** • Browse Functionality • Search Functionality • Semantic Search Functionality • Ontology Schema Editor Functionality • Ontology Instances Editor Functionality • Ontology Repository Functionality **Type of Application** • Portal • Semantic Wiki **Type of Integration** • Manual Integration Functionality

Applications		
7	MuseumFinland	**Functionality** • Browse Functionality • Generation of Navigational Views Functionality • Search Functionality • Semantic Search Functionality • Access through Diverse Devices Functionality • Multimedia Metadata Functionality • Semantic Recommender Policy Functionality **Type of Application** • Portal • Instance of a Framework **Type of Integration** • Wrappers and Mediators Integration Functionality
8	Knowledge Management Platform (KmP)	**Not considered**
9	pOWL	**Not considered**

Applications		
10	Semantic Portal of International Affairs	**Functionality** • Browse Functionality • Generation of Navigational Views Functionality • Search Functionality • Semantic Search Functionality • Access through Diverse Devices Functionality • Support for Diverse Languages Functionality • Multimedia Generation Functionality **Type of Application** • Portal **Type of Integration** • Wrappers and Mediators Integration Functionality
11	Unspecified Ontology (UNSO)	**Functionality** • Search Functionality • Semantic Query Expansion Functionality • Ontology Schema Editor Functionality • Ontology Instances Editor Functionality • Ontology Repository Functionality **Type of Application** • Ontology Tool • Semantic P2P Application **Type of Integration** •
12	Semantic Web Assistant	**Not considered**
13	Swoogle	**Not considered**

<table>
<tr><td colspan="2">Applications</td><td></td></tr>
<tr><td>14</td><td>Flink</td><td>Functionality
Browse Functionality
Multimedia Generation Functionality
Type of Application
Portal
Type of Integration
Wrappers and Mediators Integration Functionality</td></tr>
<tr><td>15</td><td>Bibster</td><td>Functionality
Browse Functionality
Search Functionality
Semantic Growth Functionality
Ontology Instances Editor Functionality
Ontology Repository Functionality
Type of Application
Portal
Instance of a Framework
Semantic P2P Application
Type of Integration
Wrappers and Mediators Integration Functionality</td></tr>
<tr><td>16</td><td>Mediator EnvirOnment for Multiple Information Sources (MOMIS)</td><td>Functionality
Browse Functionality
Search Functionality
Type of Application
Portal
Instance of a Framework
Type of Integration
Wrappers and Mediators Integration Functionality</td></tr>
</table>

Applications		
17	Annotea Shared Bookmarks	**Functionality** • Access through Diverse Devices Functionality • Ontology Instances Editor Functionality • Ontology Repository Functionality **Type of Application** • Ontology Tool **Type of Integration** • Manual Integration Functionality
18	GOHSE	**Functionality** • Dynamic and Semantic Linking Hypertext Structures Functionality • Semantic Query Expansion Functionality **Type of Application** • Instance of a Framework **Type of Integration** • Wrappers and Mediators Integration Functionality

7 Semantic Web Challenge 2005 Applications

Up until the time of this writing, SWC 2005 does not have a final report published yet. The additional goal was to show the benefit of re-using ontologies, schemas, or models [Visser & Klein, 2005]. In the 2005 challenge there was also an informal objective that was how you (the submitter) would explain the Semantic Web to your grandparents. SWC 2005 had 7 submissions presented in the next sections and summarized in Section 7.8 (Table 9).

7.1 Pytypus

In the description of the Pytypus submission to the SWC, the developers assert that Pytypus is a collaborative semantic engine that uses RDF as a base technology as most of the functionalities are described through the meaning of RDF annotations. However, at the time of this writing we could not found any publication about Pytypus. Therefore, we preferred not to consider this application on the SWC applications domain analysis.

7.2 Web Services Execution Environment (WSMX)

The Web Service Execution Environment (WSMX) [Moran *et al.*, 2005] makes available a framework for the discovery, selection, mediation and invocation of Semantic Web services. That is, WSMX provides the middleware that permits requesters and providers of Web services to find and communicate between them supported by the semantic descriptions of their functional (offerings) and non-functional (requirements and constraints on their offerings) properties. WSMX has its foundation on the conceptual model provided by the Web Services Modeling Ontology (WSMO) [Roman *et al.*, 2004] which describes various aspects related to Semantic Web services. WSMO descriptions are represented using the Web Services Modeling Language (WSML) [Bruijn, 2005].

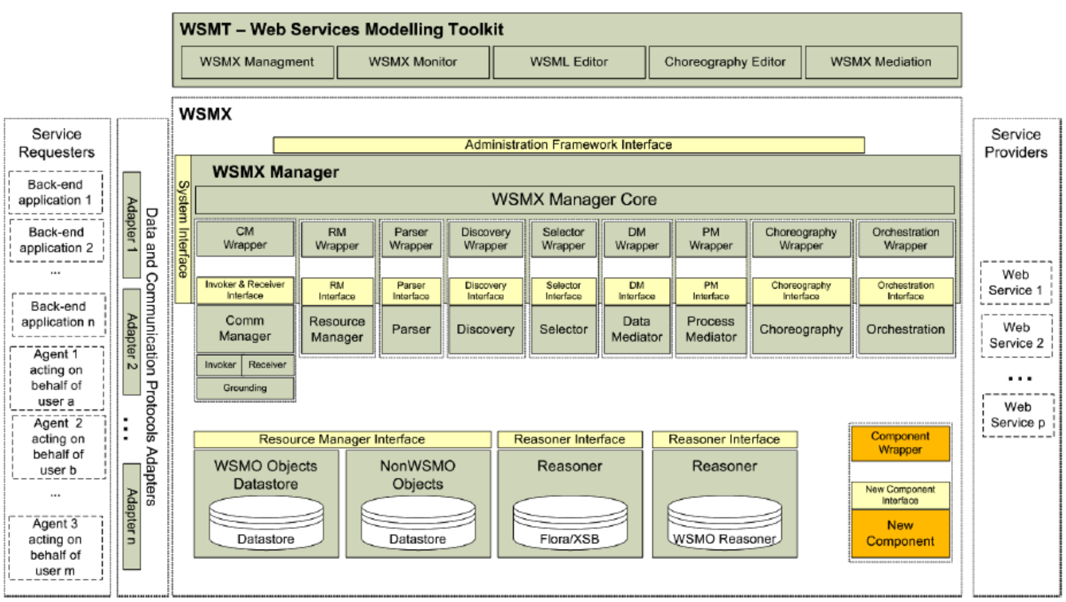

Figure 33 - WSMX Architecture [Moran *et al.*, 2005]

For a comprehensive description of the functionality of WSMX components (Figure 33), please refer to [Zaremba & Moran, 2005]. Below, a short description of the key components borrowed from [Moran *et al.*, 2005] is presented:

- The WSMX Manager Core manages the events engine, internal workflow engine and the loading of registered components at start-time;
- The Communication Manager is responsible for dealing with all aspects of sending and receiving messages to and from WSMX;
- The Resource Manager manages the persistent storage of both WSMO and non-WSMO entities;
- The Parser component parses WSML documents into equivalent WSMO4J[57] objects;
- The Discovery component is responsible for finding Web services whose capability matches the goal provided by the service requester;
- Where multiple candidate Web services are identified, the Selector component selects the Web service that provides the best match for the goal based on service requester preferences;
- During discovery or service execution, the Data Mediator or Process Mediator may be required to mediate between data and behavior from heterogeneous sources;
- The Choreography component manages the conversation between WSMX and Web services while the Orchestration component deals with the creation of new services based on the composition of existing ones;
- Reasoning support is provided using Flora-23;
- The Web Service Modeling Toolkit (WSMT) [Kerrigan, 2005] is a framework for the deployment of graphical administrative tools, which can be used with WSMO, WSML and WSMX.

The Web Services Execution Environment's domain is Semantic Web Services. This domain is broad-ranging and it is also out of the scope of this thesis. As the SWC's organizers recognized, for some applications in the 2004 edition [Klein & Visser, 2005], WSMX is an "infrastructure application" because it does not provide functionalities to the end-user. Therefore, we preferred not to consider this application on the SWC applications domain analysis. On the other hand, the organizers also acknowledge that this kind of application is of extreme importance to the Semantic Web to become more widespread.

7.3 DynamicView

DynamicView is a Semantic Web application for researchers to query, browse and visualize the distribution and the evolution of research areas in computer science. Present and past Web pages of top 20 universities in USA and China are analyzed, and research areas of faculties in Computer Science are extracted automatically by a segmentation based algorithm [Gao *et al.*, 2005].

[57] WSMO4J - http://wsmo4j.sourceforge.net/ - accessed:16/06/2006.

Different ontologies from the Association for Computing Machinery (ACM) (ACM Computing Classification System[58]) and China's Ministry of Science and Technology (MST) classification systems (classification and code of disciplines GB/T 13745/92) are combined by Simple Knowledge Organisation System (SKOS) vocabularies. Query results including numbers of researchers and their locations are visualized in SVG maps and animations [Gao *et al.*, 2005].

The major components of DynamicView are [Gao *et al.*, 2005]:

- Crawler: Faculty lists are found by humans and the Crawler searches and stores the homepage of each faculty by link analysis;
- Extraction Engine: English pages are processed automatically. Chinese pages are processed by hand due to its complexity. Extraction results in research areas, names of researchers and universities are stored in relational databases;
- Ontology Learner: The developers use the ACM digital library[59] to learn classification of research areas. Each research area is input as a keyword, and top sixty (60) papers returned with primary and additional classifications are used as training samples;
- Query Processor: Users can query by country (USA or China), ontology (ACM or MST), hot topics and history.

DynamicView is a Portal.

DynamicView uses a Wrappers and Mediators Integration functionality.

DynamicView offers the following functionalities:

- Browse functionality;
- Search functionality;
- Semantic Search functionality
- Multimedia Generation functionality; and
- Support for Diverse Languages functionality.

7.4 Personal Publication Reader (PPR)

Personal Publication Reader (PPR) is an instance of the Personal Reader Framework [Henze & Herrlich, 2004] [Henze & Kriesell, 2004]. The Personal Reader Framework is an environment for designing, implementing and maintaining personal Web content Readers [Henze & Herrlich, 2004] [Henze & Kriesell, 2004]. These personal Web content Readers allow a user to browse information (the Reader part), and to access personal recommendations and contextual information on the currently regarded Web resource (the Personal part) [Baumgartner *et al.*, 2005].

PPR makes use of Web data extraction techniques, reasoning about ontological knowledge and metadata description of information, and provides a personal semantic view on publication data [Baumgartner *et al.*, 2005]. PPR is composed by a framework (Figure 34) of Web services [Baumgartner *et al.*, 2005] for:

---

[58] ACM Computing Classification System - http://www.acm.org/class/1998/ccs98.html - accessed: 16/06/2006.

[59] ACM digital library - http://portal.acm.org/dl.cfm - accessed: 16/06/2006

- constructing the user interface;
- mediating between user requests and currently available personalization services;
- user modeling; and
- offering personalization functionality.

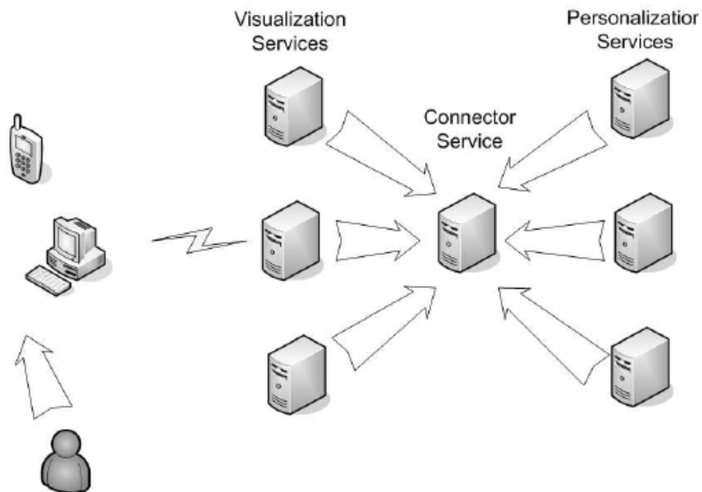

Figure 34 - Architecture of the Personal Reader framework …[60] [Baumgartner *et al.*, 2005].

PPR application demonstrates how to provide personalized, syndicated views on distributed Web data using Semantic Web technologies. The application comprises four steps [Abel *et al.*, 2005]:

- The information gathering step, in which information from distributed, heterogeneous sources is extracted and enriched with machine-readable semantics;
- The operation step for timely and up-to-date extractions;
- The reasoning step in which rules reason about the created semantic descriptions and additional knowledge bases like ontologies and user profile information; and
- The user interface creation step in which the RDF descriptions resulting from the reasoning step are interpreted and translated into an appropriate, personalized user interface.

PPR is a Portal and an Instance of a Framework.

PPR uses Wrappers and Mediators Integration functionality.

---

60 Continuation of the caption: ", showing the different components of the Personal Reader: visualization, personalization, and the Personal Reader backbone (consisting of the connector service which organizes the communication and matching between the various visualization and personalization services)" [Baumgartner *et al.*, 2005]

PPR offers the following functionalities:

- Browse functionality;
- Semantic Query Expansion functionality through its offering of personal recommendations and contextual information;
- Generation of Navigational Views functionality;
- Ontology Instances Editor functionality; and
- Ontology Repository functionality.

7.5 Oyster

Oyster[61] is a P2P application that makes use of Semantic Web techniques with the purpose of providing an infrastructure for exchanging and re-using ontologies. To accomplish this, Oyster implements a proposal for ontology metadata standard, so called Ontology Metadata Vocabulary[62] (OMV) [Palma & Haase, 2005].

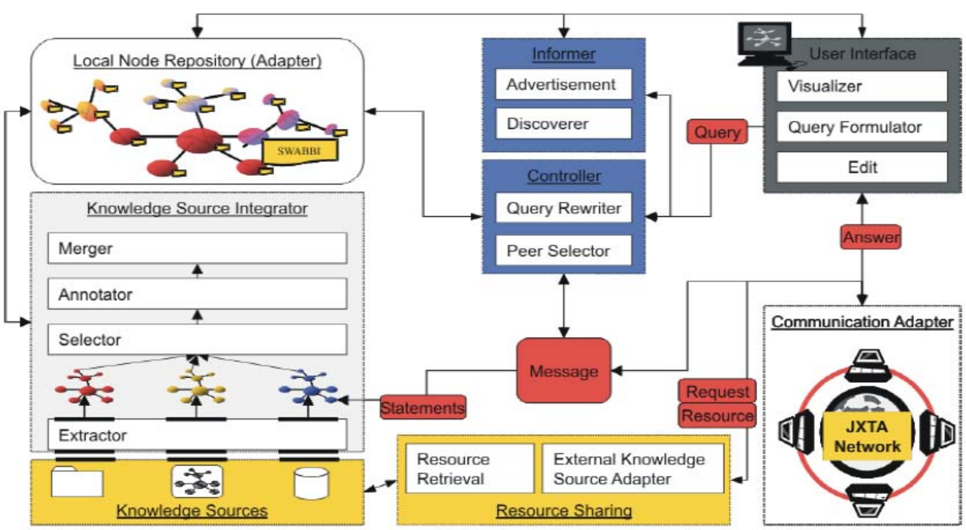

Figure 35 - Abstract Architecture of a SWAP Node [Ehrig *et al.*, 2003]

The Oyster system has been implemented as an instance of the Swapster [**SWAP EU IST-2001-34103 Final Report**, 2004] system architecture (Figure 35). In Oyster, ontologies are used in order to provide its main functions: importing data, formulating queries, routing queries and processing answers.

Oyster is a Portal, a Semantic P2P application, an Ontology Tool and an Instance of a Framework.

Oyster seems to use only one integrated ontology that is defined by the domain of ontology metadata and DMOZ Topic Hierarchy[63]. However, Oyster seems to use a kind of Wrappers and Mediators Integration functionality for integrating local knowledge sources.

---

[61] Oyster - http://oyster.ontoware.org/ - accessed: 16/06/2006

[62] OMV - http://ontoware.org/projects/omv - accessed: 16/06/2006

[63] DMOZ Topic Hierarchy - http://dmoz.org – Accessed: 16/06/2006

Oyster offers the following functionalities:

- Ontology Schema Editor functionality;
- Ontology Instances Editor functionality;
- Ontology Repository functionality;
- Browse functionality;
- Search functionality; and
- Semantic Growth functionality through its semantic duplicate detection.

7.6 FungalWeb

FungalWeb is a formal ontology design and implementation case study which provides the core for a Semantic Web application in the area of fungal genomics [Shaban-Nejad *et al.*, 2004]. The Semantic Web system can be used by human, bioinformatics applications or intelligent systems for ontology-based information retrieval to provide extended interpretations and annotations [Shaban-Nejad *et al.*, 2005].

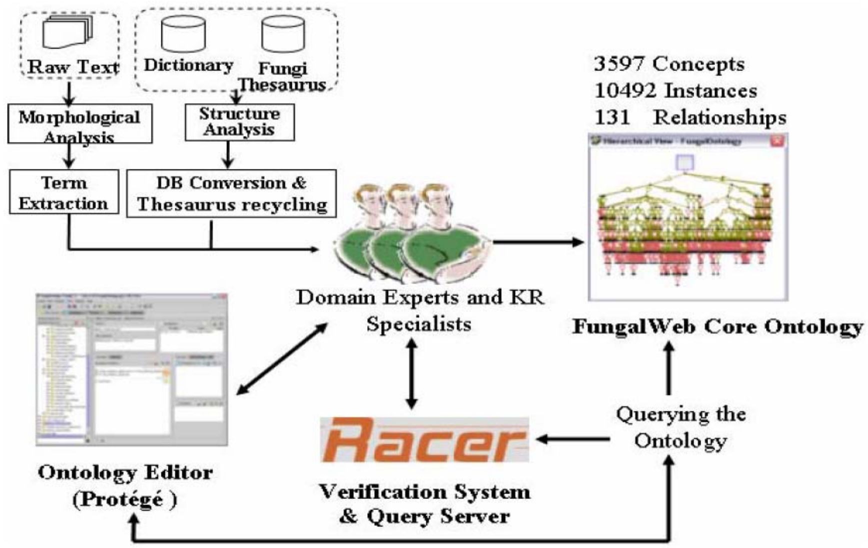

Figure 36 - Ontology Development…[64][Shaban-Nejad *et al.*, 2004].

In Figure 36 the design and development of the FungalWeb Ontology is shown and it can be divided into the following macro-steps [Shaban-Nejad *et al.*, 2004]:

- Specification;
- Knowledge acquisition;
- Implementation; and
- Semantic query.

[64] Continuation of the caption in [Shaban-Nejad *et al.*, 2004] : FungalWeb: "Ontology, the Semantic Web an Intelligent Systems for Genomics" aims to represent and map fungal genomics information using ontologies.

Figure 36 have some obsolete numbers of FungalWeb Core ontology which are updated in [Shaban-Nejad *et al.*, 2005]. According to [Shaban-Nejad *et al.*, 2005], the ontology contains 3667 concepts, 12686 instances and 157 properties; and the efforts to expand the conceptualization were continuing.

Also according to Figure 36, FungalWeb final user interface seems to be Protégé and Racer; however, Ontoligent Interactive Query Tool (OntoIQ) can also be downloaded[65] from the project's homepage. OntoIQ provides non-specialists with mechanisms to build DL-based query syntax. OntoIQ queries ontologies written in the OWL through a connection to Racer.

FungalWeb seems to use only one ontology: FungalWeb core ontology, which is obtained by a human assisted process of integration.

FungalWeb seems to offer only the Semantic Search functionality.

7.7 CONFOTO

CONFOTO[66] is a browsing and annotation service for conference photos. CONFOTO offers both an end-user-oriented browsing and editing front-end for rich annotations; and also a W3C-compliant interface to an RDF-based data store. It supports the Semantic Web idea by allowing resource descriptions to be imported, created, annotated, combined, exported, and re-purposed [Nowack, 2005].

CONFOTO implements a set of wrappers to make possible to import photo and conference data from diverse input formats, for example [Nowack, 2005]:

- RSS 2.0 feeds from w3photo - A Semantic Photo History of the IW3C2 Conferences[67];
- Atom feeds from Flickr™ [68]; or
- Proprietary XML documents from events such ESWC 2005 and XTech 2005.

The system can generate and enhance RDF data for:

- Uploaded pictures;
- Image files linked via Web-accessible URLs; and
- Photos described in external RDF/XML documents.

CONFOTO was suited for information about conferences and photos. Nevertheless, the RDF model tolerates any resource description to be combined with related objects (a FOAF file or a list of publications could be associated with a person depicted in a photo).

---

[65] OntoIQ Download form - http://www.cs.concordia.ca/FungalWeb/OntoIQ.html - accessed: 16/06/2006

[66] CONFOTO - http://www.confoto.org/ - accessed: 16/06/2006.

[67] w3photo - http://w3photo.org/ - accessed: 16/06/2006.

[68] Flickr™ - http://flickr.com/ - accessed: 16/06/2006.

The tools and features, at the time of writing, offered at confoto.org [Nowack, 2005]:

- Image Upload or Linking;
- Photo Browser;
- Annotators;
- Data Export for Re-Use.

More information about this application is necessary, but we could not find more publications about it at the time of this writing.

CONFOTO is a Portal.

CONFOTO uses Wrappers and Mediators Integration functionality.

CONFOTO offers the following functionalities:

- Browse functionality;
- Search functionality;
- Multimedia Metadata functionality;
- Access through Diverse Devices functionality;
- Ontology Instances Editor functionality; and
- Ontology Repository functionality.

7.8 SWC 2005 Summary

In Table 9, we summarize the applications submitted to the 2005's challenge; the functionalities they offer; their types and the type of integration they use. In the previous sections, we presented a brief explanation about each SWAPp as well as the functionalities they offer; their types and the type of integration they use.

Table 9 - SWC 2005 Summary

<table>
<tr><td colspan="2">Applications</td><td></td></tr>
<tr><td>1</td><td>Pytypus</td><td>Not considered</td></tr>
<tr><td>2</td><td>Web Services Execution Environment</td><td>Not considered</td></tr>
<tr><td>3</td><td>DynamicView</td><td>Functionality
• Browse Functionality
• Search Functionality
• Semantic Search Functionality
• Support for Diverse Languages Functionality
• Multimedia Generation Functionality
Type of Application
• Portal
Type of Integration
• Wrappers and Mediators Integration Functionality</td></tr>
</table>

Applications		
4	Personal Publication Reader	**Functionality** • Browse Functionality • Generation of Navigational Views Functionality • Semantic Query Expansion Functionality • Ontology Instances Editor Functionality • Ontology Repository Functionality **Type of Application** • Portal • Instance of a Framework **Type of Integration** • Wrappers and Mediators Integration Functionality
5	Oyster	**Functionality** • Browse Functionality • Search Functionality • Semantic Growth Functionality • Ontology Schema Editor Functionality • Ontology Instances Editor Functionality • Ontology Repository Functionality **Type of Application** • Portal • Ontology Tool • Instance of a Framework • Semantic P2P Application **Type of Integration** • Wrappers and Mediators Integration Functionality

Applications		
6	FungalWeb	**Functionality** • Semantic Search Functionality **Type of Application** • **Type of Integration** • Manual Integration Functionality
7	CONFOTO	**Functionality** • Browse Functionality • Search Functionality • Access through Diverse Devices Functionality • Multimedia Metadata Functionality • Ontology Instances Editor Functionality • Ontology Repository Functionality **Type of Application** • Portal **Type of Integration** • Wrappers and Mediators Integration Functionality

8 A Semantic Web Application Framework (SWAPpFW)

Our main influence to define a Semantic Web Application Framework (SWAPpFW) came from a system engineering's approach defined in [Sommerville, 2000]. Consequently, in this chapter, we define what are the framework's requirements and its architecture. We also discuss how to make the transition from the architecture to the design of the framework.

8.1 Requirements

As explained before, our framework is in the context of the SWC. Thus, the main source of requirements comes from the definition of what is a SWAPp for the challenge (Chapter 3). Therefore, in the next section, we classify the definition of a SWAPp in terms of functional and non-functional requirements.
During the review of the applications (Chapters 5 , 6 and 7), it became clear that some applications followed a "common" process for dealing with metadata. This process helps to make clear what are the main phases that could be followed by the applications. We consider this process as a source for the requirements definition as well and the process is defined in Section 8.1.2 .

The SWC applications domain analysis also is an important source of requirements because it identified the applications functionalities and how some of those functionalities could be grouped and identified as an application type (Chapter 4). Another source of requirements is the discussion about the controversialism about the Semantic Web stack (Section 2.3). We preferred not to include these discussions on the requirements since they are more related to an architectural point of view. Then, these discussions are spread on the subsections of Section 8.3 .

8.1.1 The SWC Requirements

As defined in Section 3.1 , the SWC defines a SWAPp based on set of application requirements and desirable qualities. In this section, we propose the classification of those requirements and desirable qualities as functional or non-functional requirements.

Functional requirements describe the functions or services offered by a system. The requirements depend on the type of software, its users and the type of system where the software is used. Non-functional requirements are constraints on the services or functions offered by the system. Examples of those constraints are timing constraints; constraints on the development process; and standards [Sommerville, 2000]. In a more pragmatic approach, we should classify all the requirements and desirable qualities from SWC as non-functional requirements. However, we adopted a standpoint to classify a number of those as functional requirements since they somehow represent services that SWAPps should offer.

Functional Requirements:

- R1. Considering the information sources of the applications, they must:
  - R1.1. be geographically distributed;
  - R1.2. have diverse ownerships - that is, there is no control of evolution;
  - R1.3. be heterogeneous (syntactically, structurally, and semantically);
- R2. Considering the open/close world option: the application must assume an open world; that is, it assumes that the information is never complete;
- R3. Considering the description of the data's meaning: the application must use some formal description.
- R4. Considering the data sources, they should:
  - R4.2. exploit both static and dynamic knowledge - for example, a combination of static ontologies and dynamic workflows;
  - R4.3. use the content of multimedia documents.
- R5. Considering users' access:
  - R5.1. access in multiple languages should be offered;
  - R5.2. access through devices other than a personal computer should be offered.

Non-functional Requirements:

- R1. Considering the information sources of the applications, they must:
  - R1.4. contain real-world data - that is, the sources must be more than toy examples.
- R4. Considering the data sources, they should:
  - R4.1. be used for other purposes or in another way than originally intended;
- R6. Considering scalability: should be scalable
  - R6.1. in terms of the amount of data used;
  - R6.2. in terms of distributed components working together.

As said before, we are not using the "traditional" viewpoint about requirements; therefore, we could classify the requirements R2, R3 and R4.2 as functional or non-functional requirements because they could be services offered by a SWAPp or restrictions on the functions or services of a SWAPp.

As presented in Chapters 5 , 6 and 7 , in each edition of the challenge there was an additional goal defined by the advisory board. The additional goals for the three editions reviewed were:

2003: Applications should integrate at least two heterogeneous XML data or information sources that the application's author did not manage and that allow different viewpoints.

2004: To show the benefits of the inference capabilities of the Semantic Web languages used by the applications.

2005: To show the benefit of re-using ontologies, schemas, or models.

2005: Also, an informal objective was how you (the submitter) would explain the Semantic Web to your grandparents.

For this work, we are not going to consider the goals of each year because we want the framework to be as general as possible. Additionally, as we are dealing with a framework, it will be possible to customize it to represent new goals that may be introduced in subsequent years.

8.1.2 The Metadata Handling Process

While reviewing the applications, it became clear that a sort of architecture were common to many applications: there were components or layers responsible for specific activities during the use of metadata [Hartmann & Sure, 2004] [Takeda & Ohmukai, 2005] [Shadbolt *et al.*, 2004] [Tummarello *et al.*, 2005] [Keller *et al.*, 2004] [Hyvönen *et al.*, 2005] [Mika, 2005a] [Haase *et al.*, 2004] [Beneventano & Bergamaschi, 2004] [Baumgartner *et al.*, 2005] [Baker *et al.*, 2006].

For example, the extended SEAL framework [Hartmann & Sure, 2004] has five conceptual layers which can be considered as knowledge workflows (Section 5.1 , Figure 6). Each layer has a specific assignment for dealing with data. The "Integration" and "Process and Publication" layers are responsible for inputting data into the "Representation" layer. In this layer, there is a "knowledge evolution" activity that can be considered as a form of reasoning applied to the knowledge repository. The "Representation" and the "Organization" layer can be considered as the responsible for storing the knowledge, "evolving" it and providing indexing and searching functionalities to it. Finally, the "Access" layer is responsible for the manipulation of the data in order to output in a convenient representation to the final user.

Another example is the organization of SemanticOrganizer (Section 6.5 , Figure 21). Keller *et al.* [Keller *et al.*, 2004] present a set of architectural components organized in layers. The "Representation & Reasoning" layer contains most of the components for entering data into the application ("Email ingestor", "Semantic Annotation" etc.). In the same layer, we found the "Semantic Repository" that stores both classes and instances; and the "Inference Engine" for reasoning over the data. At the "Interface" layer, we find different forms of using and accessing the data on the repository.

We believe that those layers or components represent phases of a metadata handling process in Semantic Web applications. Moreover, we propose a generalization of the layers or components into phases for which metadata have to go through in order to be offered to the end-users.

Any information system has to gathers data, stores it and makes some processing of it to offer an output to end-users. The SWAPps are somehow different from ordinary information systems because they deal with metadata that has it semantics well defined. Therefore, in any of the phases (gathering, storage and usage) the metadata manipulation can lead to the lost of semantics. Preserving the semantics is important all over the process.

In Figure 37, we present the 3 phases of the metadata handling process. The phases are Metadata Gathering, Metadata Storage and Metadata Usage. Between all the phases, there is at least one metadata flow. The metadata flow indicates that metadata that is an output of a phase serves as input to the other phase.

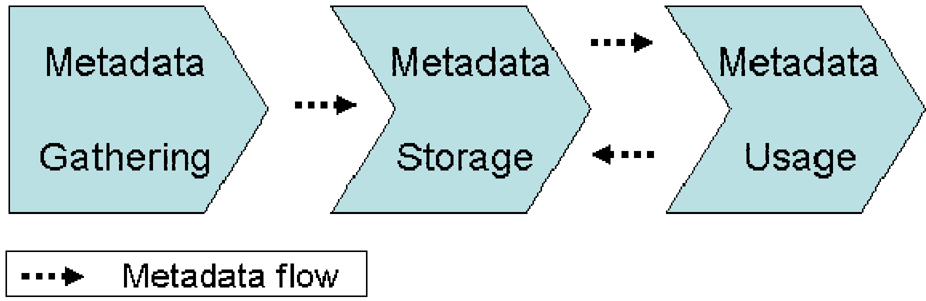

Figure 37 - The Metadata Handling Process

In the representation of the metadata flow from the Metadata Usage to the Metadata Storage, we consider that the use of metadata can generate new metadata that may have to be stored. Some applications consider that flow of metadata as a "self-flow" of the Metadata Storage phase [Hartmann & Sure, 2004] [Keller *et al*., 2004] [Mika, 2005a]. We prefer to separate it because that will depend on heuristics chosen for inferences and for updates of the metadata already gathered and stored.

Considering the last paragraph, we could simplify the metadata handling process to have only two phases: Metadata Storage and Metadata Usage. However, the existence of the Metadata Gathering phase is important because the application usefulness is directly related to the amount of information available to the user. Moreover, the development of components that are able to annotate data and give the first steps in behalf of the user is not only a desirable quality of SWAPps but also a mean to incentive the use of such applications.

We, consequently, expect that the SWAPps generated by our framework are able to have at least one component for each of the phases of Metadata Handling Process. Moreover, we expect the applications to use a common ontology (or model) during the process of handling metadata in order not to loose any semantics about the metadata. That could also be done by mechanisms that would prevent the lost of semantics, for example, if data needs to be converted from one ontology to another, or if the representation language has to be changed, then some type of annotation may be used to keep semantics that may be lost in the conversion or translation.

The SWC requirements presented on the previous section and the Metadata Handling process serves as an initial input for the definition of the architecture of our framework in the next section. As we shall see, other inputs are important for the definition of the architecture too. They are the SWC applications domain analysis, presented on Chapter 4 , and the controversialism about the Semantic Web Stack, presented on Section 2.3 .

8.2 SWAPpFW Architecture

A software architecture is, often, represented by only one diagram or document. Nevertheless, this diagram or document is not able to communicate all the concerns that a software engineer have to take into account in order to map requirements to design or deployment models. In this work, inspired by Conallen [Conallen, 1999], we take the approach of using multiple viewpoints in order to define the architecture of our framework. Connallen's introduction to visualizing an architecture through views was due to the work of Kruchten: The "4+1" View Model of Software Architecture [Kruchten, 1995]. In the next section, we present the work of Kruchten in order to use it as basis for the definition of the architecture of our framework in the following sections.

8.2.1 The "4+1" View Model of Software Architecture

The "4+1" View Model of Software Architecture is an attempt to abstract, decompose and compose with style and esthetics the design and implementation of the high-level structure of a software. To describe a software architecture, the "4+1" View Model provides a model composed of multiple views or perspectives that are presented on Figure 38 [Kruchten, 1995].

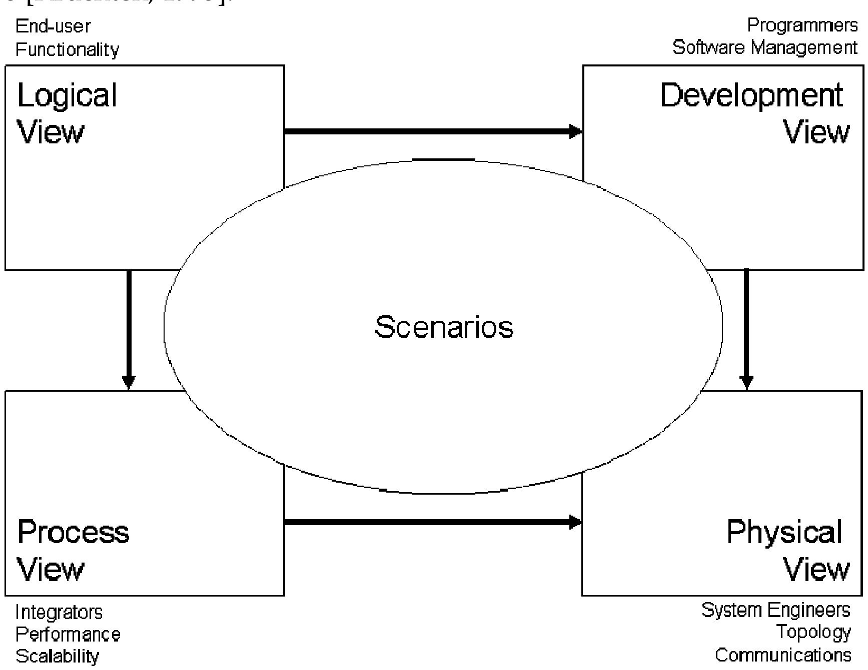

Figure 38 - The "4+1" View Model [Kruchten, 1995]

The views or perspectives on Figure 38 are [Kruchten, 1995]:

- The logical view, which is primarily the representation of the functional requirements of the software;
- The process view, which captures the concurrency and synchronization aspects of the design;
- The physical view, which describes the mappings of the software onto the hardware and reflects its distributed aspect;

- The development view, which describes the static organization of the software in its development environment;
- A few use cases, or scenarios describing the architecture.

Kruchten [Kruchten, 1995] states that the "4+1" view model is rather "generic": he defines notations for each of the views, but other notations can be used. Moreover, it is also possible to tailor the model, since not all software need all the views. A view can be omitted and others may be created if the software's context requires so. That is what we show on the next section: how we tailored the "4+1" view model so we could work with UML diagrams and represent the architecture of a Semantic Web Application framework.

8.2.2 The "4+1" View Model Tailoring

In Figure 39, we present the tailored 4+1 view model. First, the logical view was renamed as analysis view. We renamed that view in order not to generate any ambiguity since several aspects of the Semantic Web are based on Logics. However, the function of the analysis view remains the same as the logical view, which is to represent the functional requirements of the software in an object oriented class diagram.

In the process view, some non-functional requirements, such as performance and availability, are taken into account. In our understanding, those non-functional requirements could be represented by an interaction diagram such as a sequence diagram with swimlanes showing which are the phases of a process that is being executed.

An object oriented class diagram will represent the development view and it will focus on the modularization of the software as well as on the internal requirements related to the software.

A deployment diagram shall represent the physical view focusing on the representation of the distribution scalability of the architecture and, maybe, others non-functional requirements.

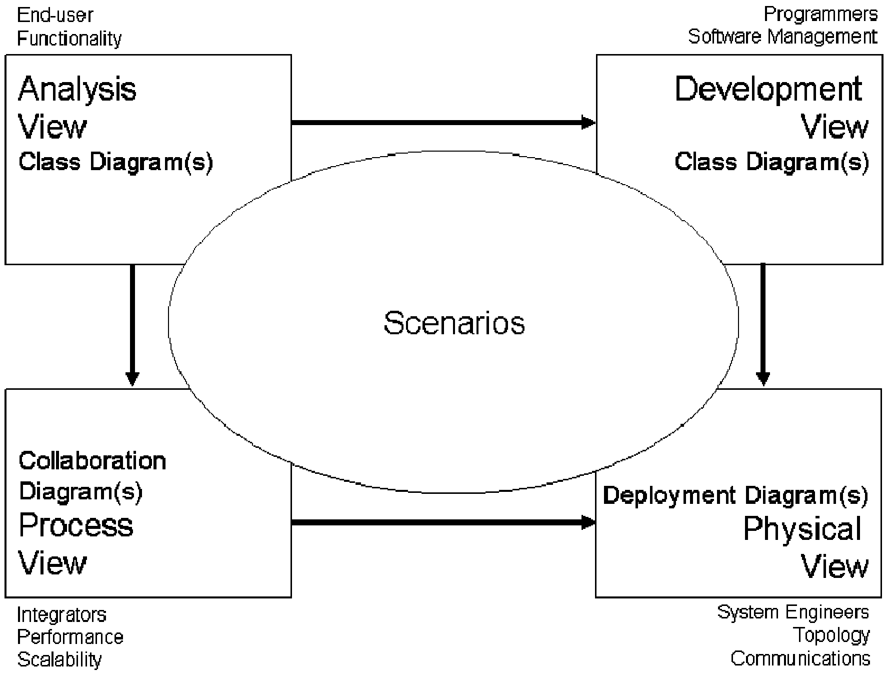

Figure 39 - The Tailored "4+1" View Model

Finally, we will represent the scenarios as use case diagrams. These scenarios shall represent all the requirements we acquired so far and any others that we may find necessary. In the next section, we present the SWAPpFW Views, that is, the instantiation of the "4+1" view model that represents the architecture of our Semantic Web Application framework.

8.3 SWAPpFW Views

In the next sub-sections, we present and briefly discuss the views of SWAPpFW. Those views are UML diagrams and represent the architecture of SWAPpFW. First, we define our scenario, followed by the presentation of the other views defined in Section 8.2.2 .

8.3.1 The Scenarios

One of most attractive characteristics of the "4+1" View Model is that different concerns are not represented, necessarily, in only one view, but they are "reconciled" by the scenarios. In Figure 40, we present the elements that compose the SWAPpFW general scenario:

- The SWC requirements (Section 8.1.1);
- The SWC applications domain analysis (Chapter 4);
- The Metadata Handling Process (Section 8.1.2); and
- The controversialism about the Semantic Web stack (Section 2.3).

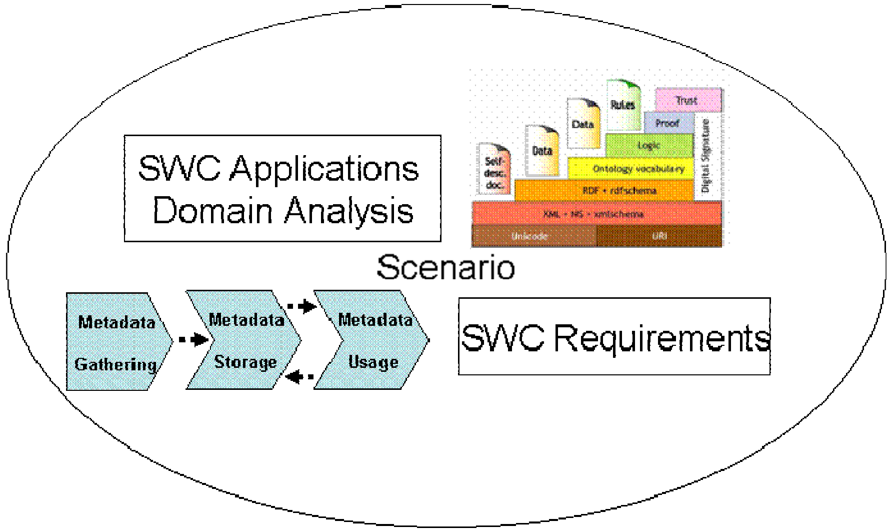

Figure 40 - SWAPpFW General Scenario

We could not, easily, map all the elements that compose the general scenario into a single use case diagram. Therefore, in the rest of this section we present a set of use case diagrams that represent those elements.

The first element that we mapped into a use case diagram was the SWC Requirements. Those requirements are quite wide-ranging. Therefore, we got a very generic use case diagram presented in Figure 41.

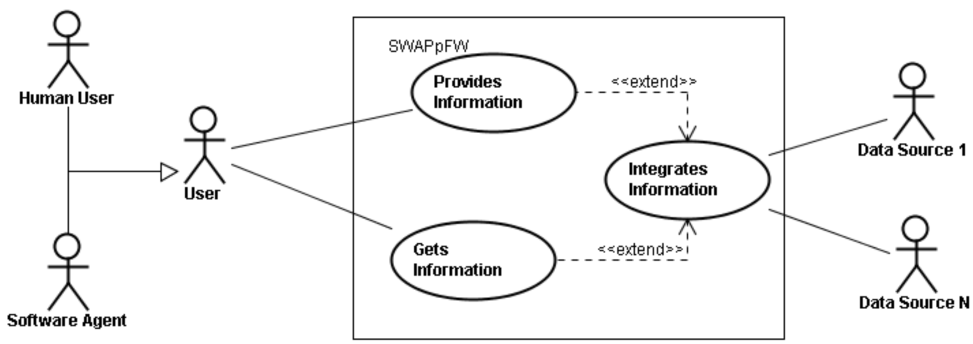

Figure 41 - Scenario 0: SWC Requirements

In the Scenario 0, there are, basically, two types of actors: Users and Data Sources. The User can be a Software Agent or a Human User. In addition, there are at least two data sources, which are necessarily heterogeneous and geographically distributed.

The use cases provided by the framework in Scenario 0 are quite generic as are the SWC requirements. They offer the user options to make use of information that may be requires the framework to integrate data from, at least, two different data sources.

We could not represent some requirements from SWC in Scenario 0. For example, the use of some formal description for the data meaning handled by the framework. However, those missing requirements shall appear in other views of the architecture. For the example given, the use of formal description for the data meaning, it will appear on the analysis view in Section 8.3.2 .

In Figure 42, Scenario 1, we incorporate a new element from the general scenario into the Scenario 0: the Metadata Handling Process. We continue with the same types of actors, and, in Scenario 1, we include new use cases: Metadata Gathering, Metadata Storage Access and Metadata Usage. Moreover, we relate those new use cases to the ones already defined in Scenario 0.

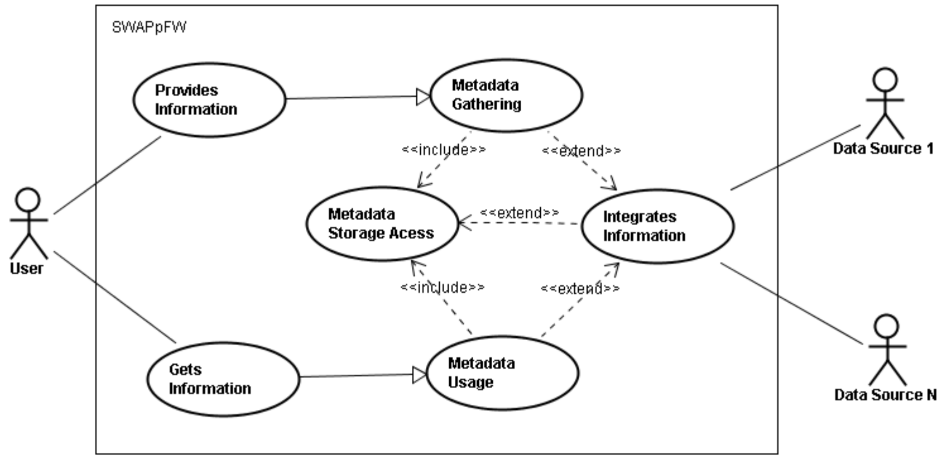

Figure 42 - Scenario 1: SWC Requirements + Metadata Handling Process

The generalizations relationships introduced in Scenario 1 mean that the Provides Information and Gets Information use cases are a kind of Metadata Gathering and Metadata Usage use cases, respectively. Additionally, the representation of the include relationship between Metadata Gathering and Metadata Usage use cases and Metadata Storage Access use case means that the first use cases will incorporate the behavior of the Metadata Storage Access use case.

The extend relationships represented in the Scenario 1 mean that: Metadata Gathering and Metadata Usage use cases, under certain conditions, will incorporate the behavior of Integrates Information use case; and, Integrates Information use case, also under certain conditions, will have to incorporate the behavior of the Metadata Storage Access use case.

Although a well-structured use case is not an overly general one [Booch *et al.*, 1998]. We chose not to map the two remaining elements of the general scenario (SWC Applications Domain Analysis and the Controversialism about the Semantic Web stack) into use case diagrams. We took that decision because those two elements are more concerned about subsystems interactions and are supposed to be transparent to end-users. Additionally, the two elements seem to be more important to the definition of the design of the framework. Nonetheless, they shall appear in the views that their influence is most perceived (Analysis view and Development view).

Now that we have defined the SWAPpFW scenario, we can go further and describe each of the other views proposed by our tailored “4+1” View Model. That is done on the next subsections.

8.3.2 The Analysis View

In the analysis view, we should keep the end-user in mind and try to show her what the functionalities of our software are. That said we present a class diagram on Figure 43 that represents the SWAPpFW.

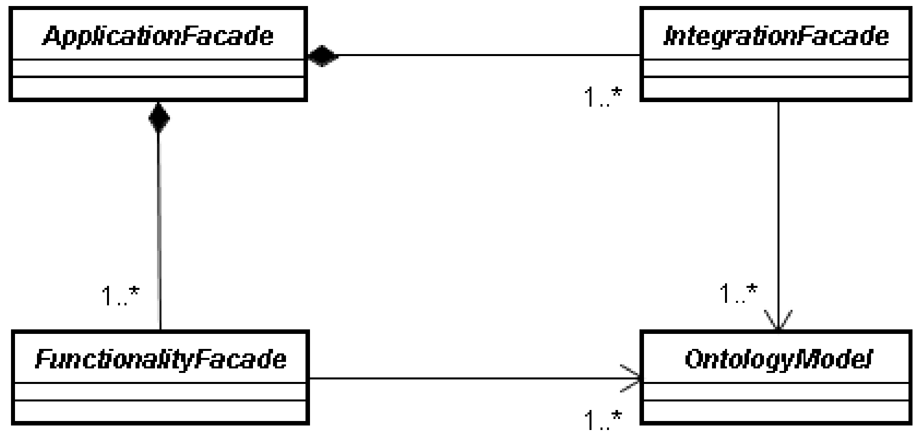

Figure 43 - SWAPpFW Analysis View Model

In the diagram of Figure 43, we considered that the types of application (ApplicationFacade) generated with SWAPpFW are composed of one or more functionalities (FunctionalityFacade) and use at least one type of data integration method (IntegrationFacade). We also considered that both the functionalities and the integration method use at least one ontology model (OntologyModel).

It is clear from the SWAPpFW analysis view model that its main influence comes from the SWC requirements. However, it also has influences from the SWC applications domain analysis, what can be seen by the use of the FunctionalityFacade class. In the development view, we go further on the definition and specialization of these general concepts presented here.

8.3.3 The Development View

The development view should serve as an artifact that will help developers to better understand and manage the static organization of the concepts, defined in the analysis view, in the software development environment. To achieve that, we go in describing and defining each one of the concepts of the analysis view model.

First, we present a class diagram for the ApplicationFacade in Figure 44. The ApplicationFacade is the Facade design pattern [Gamma *et al.*, 1995] applied to the most common types of application found in the SWC applications domain analysis.

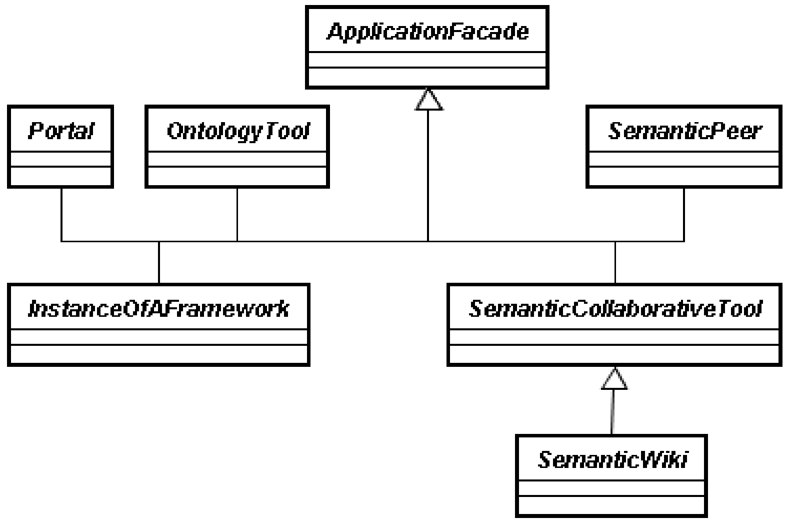

Figure 44 - Application Package

The definitions for each kind of application, shown in Figure 44, are given in Section 4.9 . Additionally, we could, orthogonally, classify the types of application by their definition or use. For example:

- The Portal and Ontology Tool are types of application that are composed of specific functionalities;
- The Instance of a Framework is a type of application where some functionalities are reused from frameworks already defined;
- The Semantic Peer is a type of application that is defined by its particular deployment model (in a Peer-to-Peer network); and
- The Semantic Collaborative Tool and its specialization, Semantic Wiki, are types of application that are defined by the process they employ while dealing with metadata.

Any of the types of application, even when defined by the composition by some functionalities, may offer “extra” ones. Moreover, the use of the Facade design pattern to represent the types of application is a way to allow the definition of other types of application. The use of the pattern also introduces the issues that each type of application is not the simple implementation of a class.

On the other hand, each type of application will be a facade that offers access to multiple classes, characterizing, in this way, a subsystem. This characterization as subsystems and its implications shall be discussed in the process view in Section 8.3.4 .

In Figure 45, we present the FunctionalityFacade, which is the Facade design pattern applied to the types of functionalities offered by the applications submitted to the SWC. As explained in the SWC applications domain analysis (Chapter 4), the functionalities here represented are those that caught our eyes because of their use of semantic or by their frequent use by the applications submitted to the challenge.

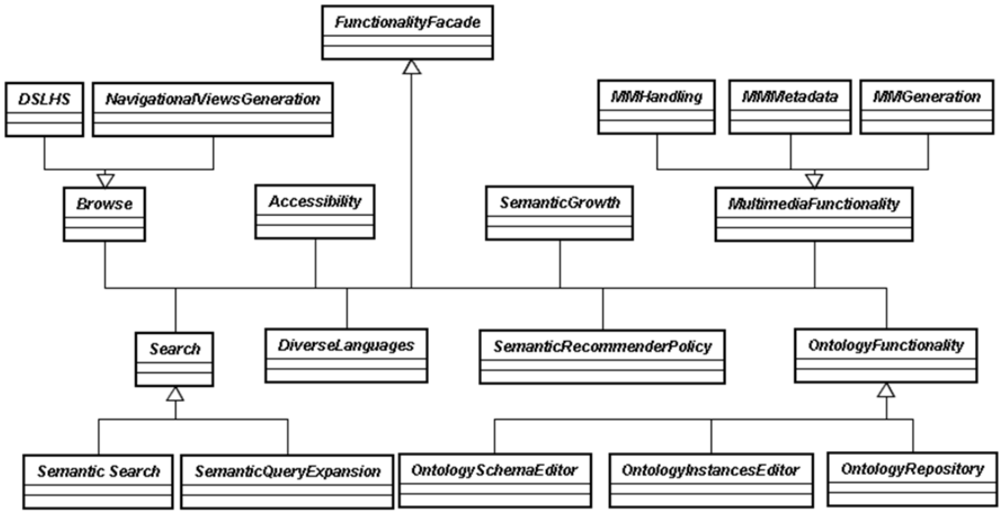

Figure 45 - Functionality Package

The definitions for each kind of functionality, shown in Figure 45, are given from Section 4.1 to Section 4.8 , totalizing 18 types of functionality. However, two functionalities are "very" abstract and were never used in the SWC applications domain analysis: Multimedia functionality and Ontology functionality. They are "only" a kind of coherent organization of the functionalities that specializes them.

An important characteristic of this set of functionalities is that all the functionalities use metadata from the application and are very generic, in the sense that they could be used by any application. The exception is the Multimedia functionality and its specializations, that may use or not metadata, and are a specific requirement of the SWC. We could, with no loss of generalization, rename the Multimedia functionality as the Document functionality. This way, we would have an even more flexible framework that could be instantiated for other domains than multimedia, e.g., Bioinformatics, Geoinformatics or a combination of them.

Any of the types of functionality may be combined to compose an application. However, at least one is necessary to characterize the application as a SWAPp. Additionally, the isolated use of some functionality or some combinations of functionalities may not make sense. As we comment on the next paragraph this is one of the inherent implications of the subsystems interaction characteristic.

The use of the Facade design pattern to represent the types of functionality is a way to allow the definition of other types of functionality. The use of the pattern also introduces the issues that each type of functionality is not the simple implementation of a class. On the other hand, each type of functionality will be a facade that offers access to multiple classes, characterizing, in this way, a subsystem. This characterization as subsystems and its implications shall be discussed in the process view in Section 8.3.4 .

One of the key requirements of the SWC was that the applications should integrate data from different data sources. In the SWC applications domain analysis, we observed that two approaches for doing that were the most representative: the wrappers and mediators integration functionality (Section 4.10.1) and the manual integration functionality (Section 4.10.2). In Figure 46, we generalize them as the IntegrationFacade class, which is the Facade design pattern applied to the type of integration.

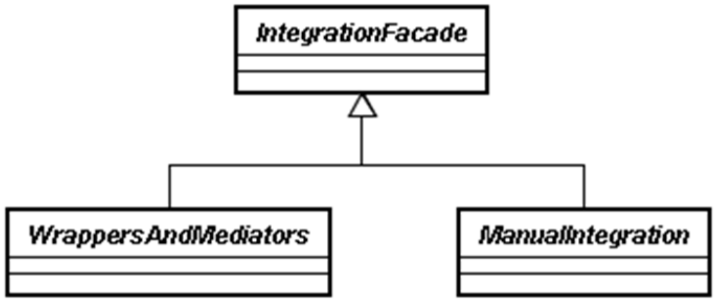

Figure 46 - Integration Package

The use of the Facade design pattern to represent the types of integration is a way to allow the definition of other types of integration. The use of the pattern also introduces the issues that each type of integration is not the simple implementation of a class. On the other hand, each type of integration will be a facade that offers access to multiple classes, characterizing, in this way, a subsystem. This characterization as subsystems and its implications shall be discussed in the process view in Section 8.3.4 .

The types of integration may be used "lonely" or they may be combined in the same application. However, by the definition of a SWAPp, any application will have at least one type of integration implemented.

Another important requirement from SWC was the use of some formal description for the data meaning of the applications. To accomplish that and to be in accordance with the Semantic Web general idea of representing semantics using ontologies, we applied the Abstract Factory design pattern [Gamma *et al.*, 1995].

The use of this design pattern allows the designer of the applications to configure them with one or more ontology factories (OntologyFactory). The first product that, naturally, came to mind was the ontology model (OntologyModel). However, as the applications evolve or the understanding about Semantic Web techniques is clearer to the designer, other products may be associated with the abstract factory as well as other factories may be created. In Figure 47, we present the abstract classes (OntologyFactory and OntologyModel) as well as two possible instantiations of those classes, one to deal with RDF models and the other to deal with OWL-DL models.

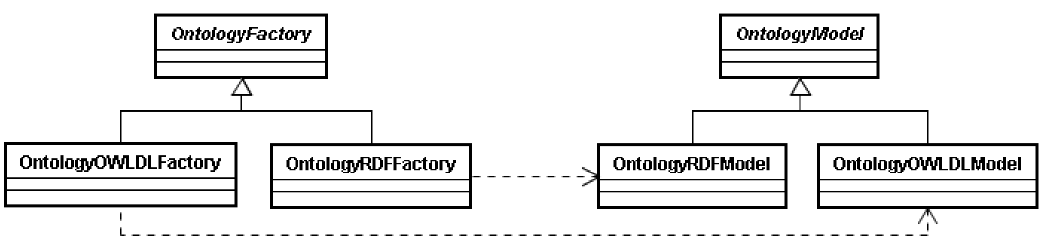

Figure 47 - Ontology Package

The use of the Abstract Factory to represent the ontology model somehow addresses the requirements about the controversialism about the Semantic Web stack since the designer can configure its application to use his understanding about the controversialism. The use of the pattern also allows for the ontology spectrum, presented in Section 2.1 , to be considered when developing an application.

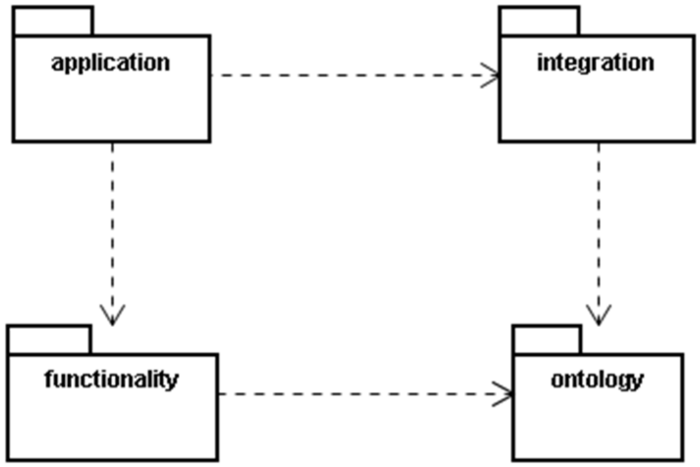

Figure 48 - Package Dependencies

Finally, we present, in Figure 48, a class diagram depicting the packages that contain the classes of each diagram already presented in this development view and how they are dependent on each other. As we have seen in this section, there are some issues about subsystems interactions that will be discussed in the next section, the process view.

8.3.4 The Process View

The process view can be described at several layers of abstraction, each level addressing a concern [Kruchten, 1995]. In our case, the process view addresses how the main entities of the analysis view fit within the process architecture. To do that, we have already seen that the Metadata Handling Process, defined in Section 8.1.2 , will have to be taken into account. That was already done when we defined the use cases in the scenarios presented in Section 8.3.1 . However, as stated in the same section, the generality of the use cases defined is not a desirable quality for well-structured use cases.

Another source of information for the process view is the subsystems interactions, as stated in the development view. The number of possible subsystems interactions may be prolific at the level of abstraction that we are dealing, the architectural. Therefore, we postpone this discussion to the design level presented in Section 8.4 .

8.3.5 The Physical View

The physical view deals with questions about the deployment of the SWAPpFW. By the SWC requirements, the requirement to have at least two distributed, non-controlled-ownership and heterogeneous information sources can be seen at this view. Additionally, from the SWC applications domain analysis we learned that there is one special type of application that attends this requirement without using a “traditional” client-server architecture, the Semantic Peer-to-Peer application.

Therefore, we present two rather general deployment diagrams that represent a Client-Server Web application (Figure 49) and a Peer-to-Peer network of applications (Figure 50).

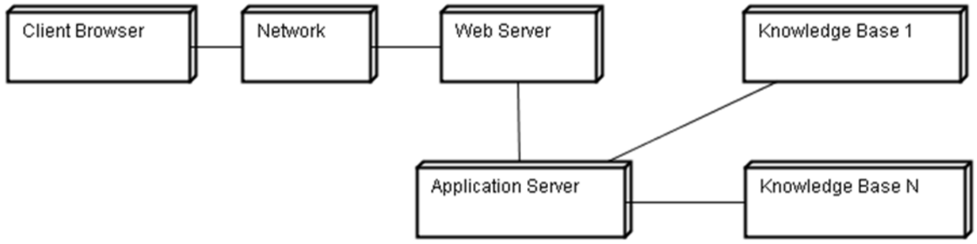

Figure 49 - Client-Server Web Application

In Figure 49, we present a client browser, which is used by the end-user to access through the network a Web server. The Web server provides access to an application server, which communicates with two knowledge bases and, probably integrates their information. We used the name knowledge base in order to be generic, and, for example, not to restrict the developer to use an ontology repository server or a database server.

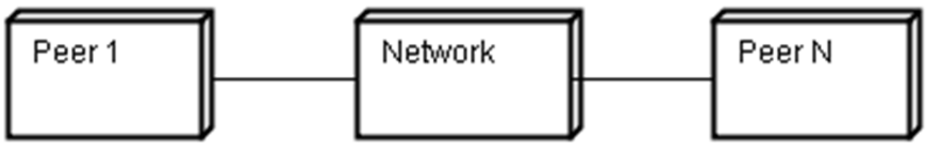

Figure 50 - P2P Network of Applications

The two deployment diagrams are not the only options that a developer has. That is one of the reasons we made them generic. Other kinds of deployment could be used to apply distribution strategies. This has to guarantee, of course, that the requirement to have at least two information sources that are distributed, non-controlled-ownership and heterogeneous is attended.

8.4 SWAPpFW Design

In this section, we discuss how to approach the transition from the SWAPpFW architecture to the SWAPpFW design. Even though the SWAPpFW domain is restricted by the SWC requirements, as we saw in the definition of the scenarios in the architecture, that restriction is still blurred and wide-ranging. Therefore, in the next section we discuss the “complexity” of the transition from the architecture to the design, or the number of possible design configurations. In the following section, we present a possible design configuration and its instantiation.

8.4.1 Possible Design Configurations (“Complexity”)

By the analysis view, we have that an application offers at least one type of functionality and at least one integration method; and both of them use an ontology model. As stated in the development view, we found in the SWC application domain analysis 18 possible types of functionality that an application could offer. However, two of the types were abstract and never used in the domain analysis. That left us with 16 possible types of functionality so far.

Any of the types of functionality could be combined to compose an application and that would be a possible design configuration. However, the isolated use of some functionality or some combinations of functionalities may not make sense. Having that number of types of functionality would then provide us with less than 2^{16} (65.536) possible combinations of functionalities. The types of application and integration methods also impose some restrictions on that number of possibilities. However, it is still a considerable amount.

There are then several possible combinations of functionalities. We identified some of them in the applications submitted to SWC. However, the developer should not be limited by them. In addition, some combinations of functionalities might not attend the definition of a SWAPp *per se*. An application to be considered a SWAPp in this work has to attend the requirements from the SWC. We have tried to make that definition more clear and precise when defining the SWAPpFW, however if it is taken out from its context, someone could imagine that getting together some functionalities as the ones defined in the SWAPpFW would them provide she with a SWAPp. That is not necessarily true; the developer might end up with an application that does not attend the requirements from SWC.

From the applications submitted to SWC, we could find some type of applications that were combination of functionalities. For example, we defined the Portal type of application as offering the browse and search functionality. In Table 10, we present the applications that were classified as Portals as well as their functionalities. If we take a closer look at the table, we can find a sub-set with 10 (ten) applications that besides offering the Search and Browse functionality also offers a Semantic Search functionality. Other sub-sets or combinations of functionalities that we can identify are the ones that make use of the Ontology Instances Editor, Ontology Repository and Multimedia Generation functionalities.

Table 10 - Portal Applications

Application **Functionality - Type of Application - Integration**	SEAL	DOPE	SECO	Building Finder	CS AKTive Space	GeoShare	MusiDB	MADIERA Portal	SemanticOrganizer	Platypus Wiki	MuseumFinland	SPIA	Flink	Bibster	MOMIS	DynamicView	PPR	Oyster	CONFOTO	
Browse	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	19
Generation of Navigational Views	1										1	1					1			4
Dynamic and Semantic Linking Hypertext Structures																				0
Search	1	1	1	1	1	1	1	1	1	1	1	1		1	1	1		1	1	17
Semantic Search	1	1		1	1	1		1		1	1	1				1				10
Semantic Query Expansion		1				1	1										1			4
Access through Diverse Devices											1	1							1	3
Support for Diverse Languages	1				1			1				1				1				5
Multimedia Handling									1											1
Multimedia Metadata				1	1	1					1								1	5
Multimedia Generation		1			1	1						1	1			1				6
Semantic Growth					1				1					1				1		4
Semantic Recommender Policy							1				1									2
Ontology Schema Editor										1								1		2
Ontology Instances Editor									1	1				1			1	1	1	6
Ontology Repository									1	1				1			1	1	1	6
Portal	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	19
Ontology Tool										1								1		2
Instance of a Framework	1	1									1			1	1		1	1		7
Semantic P2P Application														1				1		2
Semantic Collaborative Tool					1															1
Semantic Wiki										1										1
Wrappers and Mediators Integration	1	1	1	1	1	1	1	1			1	1	1	1	1	1	1	1	1	17
Manual Integration									1	1										

Table 11 presents the applications classified as Ontology Tools. An ontology tool, as defined before, should offer the Ontology Instances Editor, Ontology Repository and Ontology Schema Editor functionalities. However, we can verify that also the Search functionality is offered by the applications presented in Table 11.

Table 11 - Ontology Tools Applications

Application **Functionality - Type of Application - Integration**	Platypus Wiki	Unspecified Ontology (UNSO)	Annotea Shared Bookmarks	Oyster	
Browse	1			1	2
Search	1	1		1	3
Semantic Search	1				1
Semantic Query Expansion		1			1
Access through Diverse Devices			1		1
Semantic Growth				1	1
Ontology Schema Editor	1	1		1	3
Ontology Instances Editor	1	1	1	1	4
Ontology Repository	1	1	1	1	4
Portal	1			1	2
Ontology Tool	1	1	1	1	4
Instance of a Framework				1	1
Semantic P2P Application		1		1	2
Semantic Wiki	1				1
Wrappers and Mediators Integration				1	1
Manual Integration	1		1		2

Other kinds of applications were not defined as combinations of functionalities. For example, the Instance of a Framework type of application, the applications of this type were supposed to instantiate or to reuse the functionalities offered by frameworks. In Table 12, we present those applications and it can be seen that the there is a "dispersion" in the functionalities used by the applications. Most of them are portals but there are also ontology tools and semantic P2P applications.

Table 12 - Instance of a Framework Applications

Functionality - Type of Application - Integration / Application	SEAL	SECO	Semblog	MuseumFinland	Bibster	MOMIS	GOHSE	PPR	Oyster	
Browse	1	1		1	1	1		1	1	7
Generation of Navigational Views	1			1				1		3
Dynamic and Semantic Linking Hypertext Structures							1			1
Search	1	1		1	1	1			1	6
Semantic Search	1	1		1						3
Semantic Query Expansion		1					1	1		3
Access through Diverse Devices				1						1
Support for Diverse Languages	1									1
Multimedia Metadata				1						1
Multimedia Generation		1								1
Semantic Growth					1				1	2
Semantic Recommender Policy			1	1						2
Ontology Schema Editor									1	1
Ontology Instances Editor			1		1			1	1	4
Ontology Repository			1		1			1	1	4
Portal	1	1		1	1	1		1	1	7
Ontology Tool									1	1
Instance of a Framework	1	1	1	1	1	1	1	1	1	9
Semantic P2P Application					1				1	2
Semantic Collaborative Tool			1							1
Wrappers and Mediators Integration	1	1		1	1	1	1	1	1	8
Manual Integration			1							1

We classified 3 applications as Semantic P2P applications, but 2 of them were instances of the same framework. That is reflected on the functionalities offered by that type of application. However, as explained before, the Semantic P2P type of application is not a type of application that is related to the grouping of some functionalities. It is more related to the way the (physical) architecture of the application was designed and how it uses metadata with the support of P2P techniques and tools.

In view of that, in the next section we describe a possible design configuration, which is a valid combination of functionalities, its instances and a possible implementation.

8.4.2 A Valid Combination of Functionalities

We could consider a type of application (T1) that offers the following functionalities:

- Browse functionality;
- Search functionality;
- Semantic Search functionality; and
- Multimedia generation functionality.

We would then classify this application as a Portal. Let us also consider that this application integrates data from researchers and their publications. If we look at the three editions of the SWC, we can find some applications that are portals, offer those functionalities and integrate those kinds of information sources:

- 2003: CS AKTive Space [Shadbolt *et al.*, 2004];
- 2004: Flink [Mika, 2005a]; and
- 2005: DynamicView [Gao *et al.*, 2005].

Some applications offer more functionalities. Others, as Flink, have some "potential" functionalities. As Flink is implemented on top of Sesame [Broekstra *et al.*, 2002], it could offer the Search and Semantic Search functionalities. Even tough each of these applications has its peculiarities, they could still be considered instances of T1 and then T1 would be a valid combination of functionalities.

An implementation of T1 would be Elmo[69]. Elmo is Java API for SWAPps. It is open source and is distributed under the LGPL license[70]. Great part of the Elmo code has originally appeared in Flink, the winner of SWC 2004.

Elmo stands on top of Sesame storage and query facility. Elmo provides support for developing SWAPps using popular ontologies, including FOAF, RSS 1.0 and Dublin Core. The Elmo object model builds on a single approach: each ontological concept has a matching Java class in the library, using the same name. Its static object model can be extended with new concepts, relationships or new ontologies. Elmo also offer some tools to work with the ontologies, for example an RDF crawler and a smusher for FOAF data [Mika, 2005b].

8.5 Summary

In this chapter, we presented a Semantic Web Application framework (SWAPpFW). First, we elicited the requirements based on the SWC requirements and desirable qualities for a SWAPp. We classified those requirements as functional and non-functional requirements. In addition, as part of the requirements elicitation, we defined the Metadata Handling Process, which we identified during the review of the applications submitted to the SWC and is a contribution of this work. It is important to reaffirm that, even not appearing as an element of the process, reasoning is part of it. We chose to represent reasoning as the interaction between the Metadata Storage phase and Metadata Usage phase. We made that choice because reasoning will depend on heuristics chosen for inferences and for updates of the metadata already gathered and stored.

After the requirements, we proposed an architecture to the SWAPpFW using a tailored version of the "4+1" view model of software architecture. The views and discussion generated during this activity are a contribution of this work as well. Based on the views we discussed how difficult it might be to go from the architecture of the SWAPpFW to its design. However, an example of a solution is presented, even tough not being a best-case solution.

---

[69] openRDF.org - http://www.openrdf.org/ - accessed: 28/10/2006

[70] GNU Lesser General Public License (LGPL) - http://www.gnu.org/licenses/lgpl.html - accessed: 28/10/2006

The requirements for the framework and the framework itself were influenced by our choice on using the SWC applications. A clear example of that are the Multimedia functionality and its sub-functionalities. The use of multimedia documents is an explicit requirement from SWC but it is still vague and wide ranging as we could see by the different functionalities derived from it. Our framework and, specially, the functionalities and types of applications rely strongly on the SWC requirements. However, the framework itself is also generic and inclusive. We could define new functionalities for dealing with other aspects of SWAPps, for example aligning ontologies, dealing with geo-referenced metadata or bioinformatics metadata.

9 Conclusion, Contributions and Future Works

For the end-users, the benefits from using Semantic Web techniques or technologies should be transparent. On the other hand, for the developers, it is important to understand how those techniques or technologies relate to each other and which decisions have to be taken in order to achieve the benefits offered by the “new” explicit semantics of data.

In view of that, in this work we presented SWAPpFW, a Semantic Web application framework. The main differential of this framework is its domain, the domain of the SWC. Also important as a differential of the SWAPpFW is the bottom-up and empirical approach to build it. Those differentials are important and played an expressive role on the definition of the framework.

The use of the SWC applications allowed us to use a specific and still broad ranging set of applications for the Semantic Web as the sources for the SWAPpFW. However, other sources and applications could have been used, for example the list of applications provided by the W3C´s Semantic Web Best Practices Working Group. If we had used other set of applications, new kinds of functionalities and types of applications might have been discovered. Nevertheless, we still believe that the framework proposed is still broad-ranging enough to accommodate other specificities such as the use of other kinds of functionalities, such as, the use of geo-referenced and bioinformatics metadata or ontology functionalities not encountered in the SWC applications as discussed before.

To come up with the SWAPpFW, one of the sources of requirements was the review of the SWC applications. We did the review using an RDF vocabulary and based on papers and articles about the applications. That review provided a deeper understanding of the applications but we did not have the chance to scrutinize every aspect of them such as the applications’ code or architecture. Sometimes because they were not available or because some projects were huge. That missing part of our analysis could have offered more elements for the framework and help it become even more specific, nevertheless it could also generate a narrow understanding of the problem to be solved by the framework.

The definition of the RDF vocabulary provides a mean to reproduce the review of the applications, but the designer of a new SWAPp should also invoke its experience, common sense, and as many sources as possible to keep the better understanding of the applications in order to extend the framework. This way, she can accommodate more functionalities into an application or define other kinds of applications to the framework. That been said, in the next sections, we, briefly, present the contributions of this work as well as the future works.

9.1 Contributions

The main contributions of this work are:

- The creation of an extended DOAP vocabulary to review the applications submitted to SWC;
- The domain analysis of the SWC applications identifying their types of application and functionalities;
- The definition of the Metadata Handling Process that represents the main phases that a SWAPp has to go through to deal with metadata and not lose the semantics of it; and
- The presentation of a Semantic Web application framework (SWAPpFW), mainly, by its architecture and using an empirical approach based on the domain analysis of the SWC applications.

9.2 Future Works

Based on what has already been done, we present some future works that need to be done. We briefly introduce them by their complexity. Firstly, there is the online publication of the extended DOAP vocabulary and of the review of the SWC applications using the extended DOAP vocabulary. Those tasks completed, we would have some input from other researchers or developers if the extension attend their needs when describing SWAPps and if any improvements could be carried out to help identifying the types of functionalities, applications and integration methods used by a SWAPp.

Another future work is the improvement of the comparison of SWAPpFW and its related work. This work would require a review of the literature to find works that are similar to the SWAPpFW and how our work could be improved. To this work, more effort and understanding of the Semantic Web and Software Engineering areas is required. Valuable contributions and improvements to the SWAPpFW can be foreseen such as incorporating different methods on how other approaches deal with reasoning when using metadata.

A more complex future work would be the refinement of the framework to better deal with the transition from the architecture to the design leading to a family of Product Line Architectures. That can be achieved by studying and understanding the related works as well as by the implementation of SWAPpFW instances. Those implementations would also guarantee the framework adequacy and relevance.

Lastly, but not less important, would be the study of the complexity and the improvement of each functionality. This future work may give ground to many other works in different areas and of diverse complexity. Aspects such as the framework dependency on the SWC requirements could be better discussed, analyzed and improved by the research of (other) functionalities such as the ontology functionality and the treatment of domain dependent metadata, for example, geo-referenced and bioinformatics metadata.

10 Glossary - Acronyms and Vocabulary

Table 13 - Acronyms

Acronym	Description
3D	Three dimensional
ACM	Association for Computing Machinery
ADTF	Applications and Demos Task Force
API	Application Programming Interface
BUSTER	Bremen University Semantic Translator for Enhanced Retrieval
CAS	CS AKTive Space
CIA	USA's Central Intelligence Agency
COHSE	Conceptual Open Hypermedia Service
CORESE	Conceptual Resource Search Engine
CS	Computer Science
CSS	Cascading Style Sheets
DAML+OIL	DARPA Agent Markup Language + Ontology Inference Layer
DARPA	USA's Defense Advanced Research Projects Agency
DDI	Data Documentation Initiative
DLS	Distributed Link Service
DOAP	Description Of A Project
DOPE	Drug Ontology Project for Elsevier
ECHO	Earth Observing System ClearingHouse
ELSST	European Language Social Science Thesaurus
ESWC 2005	The Second European Semantic Web Conference - 2005
FOAF	Friend Of A Friend RDF/XML vocabulary.
GAV	Global As View
GCMD	Global Change Master Directory
GO	Gene Ontology
GOHSE	GO + COHSE
GVV	Global Virtual View
HTML	Hypertext Markup Language
HTTP	HyperText Transfer Protocol

Acronym	**Description**
INRIA	Institut National de Recherche en Informatique et en Automatique
IR	Information Retrieval
ISWC2002	International Semantic Web Conference (ISWC) 2002
ISWC2003	The Second International Semantic Web Conference
ISWC2004	The Third International Semantic Web Conference
ISWC2005	The Fourth International Semantic Web Conference
IW3C2	International World Wide Web Conference Committee
JSP	Java Server Pages
KmP	Knowledge Management Platform
KMS	Knowledge Management System
MADIERA	Multilingual Access to Data Infrastructures of the European Research Area
MDA	Model Driven Architecture
MLH	Multi-Layered Hypercube
MOMIS	Mediator EnvirOnment for Multiple Information Sources
MST	A classification and code of disciplines GB/T 13745/92 by Ministry of Science and Technology, China
NASA	USA's National Aeronautics and Space Administration
NESSTAR	Networked Social Science Tools and Resources
OBA	Ontology-Based Application
ODL	Object Definition Language
ODL-I3	A language that extends ODL with an underlying Description Logic
ODMG	Object Database Management Group
OGC	Open Geospatial Consortium
OHS	Open Hypermedia System
OMG	Object Group Management
OMV	Ontology Metadata Vocabulary
OntoIQ	Ontologent Interactive Query Tool
OWL	Web Ontology Language
OWL DL	A specie (DL) of Web Ontology Language (OWL)
OWL Full	A specie (Full) of Web Ontology Language (OWL)

Acronym	Description
OWL Lite	A specie (Lite) of Web Ontology Language (OWL)
P2P	Peer-to-peer
Perl	Practical Extraction and Report Language
PHP	Hypertext Preprocessor
PPR	Personal Publication Reader
PRF	Personal Reader Framework
PROLOG	PROgramming in LOGic
RDF	Resource Description Framework
RDF Schema	RDF Vocabulary Description Language 1.0
RDQL	RDF Data Query Language
RSS	RDF Site Summary (RSS 0.9 and 1.0)
RSS	Rich Site Summary (RSS 0.91, RSS 1.0)
RSS	Really Simple Syndication (RSS 2.0)
RSS	Real-time Simple Syndication (RSS 2.0)
SEAL	SEmantic portAL
SECO	SEmantic COllaboration
SHOE	Simple HTML Ontology Extensions
SKOS	Simple Knowledge Organisation System
SPIA	Semantic Portal of International Affairs
SVG	Scalable Vector Graphics
SW	Semantic Web
SWAP	Semantic Web Accessibility Platform
SWAPp	Semantic Web Application
SWAPSA	Semantic Web and Peer-to-Peer Project - System Architecture. See Swapster.
Swapster	A generic platform to account for the general need of sharing semantic-based information in P2P fashion. See SWAPSA.
SWBPD	Semantic Web Best Practices and Deployment Working Group
SWC	Semantic Web Challenge
SWC 2003	Semantic Web Challenge 2003
SWC 2004	Semantic Web Challenge 2004
SWC 2005	Semantic Web Challenge 2005

Acronym	Description
SWD	Semantic Web Documents
SWEET	Semantic Web for Earth and Environmental Terminology
SWWS'01	Semantic Web Working Symposium (SWWS) 2001
UML	Unified Modeling Language
UNSO	UNSpecified Ontology
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
WCAG	Web Content Accessibility Guidelines
wff	Well formed formula
WSML	Web Service Markup Language
WSMO	Web Service Modeling Ontology
WSMT	Web Service Modeling Toolkit
WSMX	Web Service Execution Environment
WWW	World Wide Web
XML	eXtensible Markup Language
XSLT	eXtensible Stylesheet Language Transformation

Table 14 - Vocabulary

Term	Description
3store	RDF Schema Triplestore
Aggregator	An aggregator or news aggregator is a type of software that retrieves syndicated Web content that is supplied in the form of a Web feed (RSS, Atom etc.)
Annotea	A Generic Annotation Environment Using RDF/XML
Annotea Ubimarks	An application of Annotea shared bookmarks in Mozilla
AnnoTerra	Annotated Terrestrial Information
Atom	Atom is an XML-based document format that describes lists of related information known as "feeds"
Bibster	A Peer-to-Peer system for exchanging bibliographic data among researchers
Brainlet	A DBin application
CONFOTO	A semantic browsing and annotation service for conference

Term	Description
	photos
Crawler	Also known as a Web crawler, Web spider or Web robot is a program that browses the World Wide Web in a methodical, automated manner.
Datalog	A subset of PROLOG. Datalog is a language of facts and rules
DOAPamine	DOAP Annotations for Java 5
E-Commerce	Electronic Commerce
eDonkey	File sharing application
Elcano Institute	Spain's Real Instituto Elcano de Estudios Internacionales y Estratégicos
eMule	File sharing application
Flickr™	An online photo management and sharing application
Google	The Google™ search engine. "Googol" is the mathematical term for a 1 followed by 100 zeros. Google's play on the term reflects the company's mission to organize the immense amount of information available on the Web.
Kazaa	File sharing application
MuseumFinland	Finnish Museum on the Semantic Web
MusicBrainz	A user-maintained community music metadatabase
OntoWeb	Ontology-based information exchange for knowledge management and e-commerce
Oyster	Peer-to-Peer application that exploits Semantic Web techniques in order to provide a solution for exchanging and re-using ontologies.
Servlet	Server + applet, a server-side application
Sesame	A Generic Architecture for Storing and Querying RDF and RDF Schema
Swoogle	According to the authors [Ding *et al.*, 2004]: "Semantic Web Ontology ..." Well, the developers are still trying to figure out the rest.
W3photo	A Semantic Photo History of the IW3C2 Conferences
WIKI	What I Know Is (a content management system, or "quick" in

Term	Description
	Hawaiian)
WSMO4J	An API and a reference implementation for building Semantic Web Services applications compliant with WSMO.
XTech 2005	Formerly known as the XML Europe conference. XTech has widened its scope to incorporate neighboring technologies from the Web and business

11 References

ABEL, F. *et al.* The Personal Publication Reader. In: GIL, Y. *et al* (Eds.). **The Semantic Web – ISWC 2005: 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland, November 6-10, 2005. Proceedings.** Lecture Notes in Computer Science. Berlin / Heidelberg Springer, v.3729, 2005. p.1050-1053.

ALAVI, M.; LEIDER, D. **Knowledge management systems: emerging views and practices from the field.** In: 32nd Annual Hawaii International Conference on System Sciences - HICSS-32. Maui, HI, USA. January, 1999. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=16787&arnumber=772754&count=68&index=8. Acessed: 2005-09-22.

ALVHEIM, A.; RYSSEVIK, J. **MADIERA - Multilingual Access to Data Infrastructures of the European Research Area.** In: Second International Conference on e-Social Science. Manchester, UK. June, 2005. Available at: http://www.madiera.net/presentations/. Acessed: 2006-03-07.

ASHBURNER, M. *et al.* Gene Ontology: tool for the unification of biology. **Nature Genetics**, v.25, n.25-29, p.334-342, May, 1st. 2000. Available at: http://www.nature.com/ng/journal/v25/n1/full/ng0500_25.html. Acessed: 2006-04-17.

AUER, S. **Powl - A Web Based Platform for Collaborative Semantic Web Development**. In: SFSW 05 - Workshop on Scripting for the Semantic Web. Hersonissos, Crete, Greece: CEUR Workshop Proceedings (CEUR-WS.org). May, 30th, 2005. Available at: http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS//Vol-135/. Acessed: 2006-03-10.

BAADER, F. *et al.* (Eds.). **The Description Logic Handbook: Theory, Implementation and Applications.** Cambridge University Press, 2003. 574 p.

BAKER, C. J. O. *et al.* Semantic Web Infrastructure for Fungal Enzyme Biotechnologists. **Web Semantics: Science, Services and Agents on the World Wide Web**, v.4, n.3, p.168-180. 2006. Available at: http://www.websemanticsjournal.org/ps/pub/2006-12. Acessed: 2006-11-10.

BARISH, G.; KNOBLOCK, C. A. **An efficient and expressive language for information gathering on the web**. In: The Sixth International Conference on AI Planning and Schedulling - AIPS - Workshop on Is there life after operator sequencing? -- Exploring real world planning. Tolouse, France: AAAI Press, 2002. Available at: http://www.isi.edu/cgi-bin/ariadne/integration/papers.cgi. Acessed: 2006-04-20.

BAUMGARTNER, R.; HENZE, N.; HERZOG, M. The Personal Publication Reader: Illustrating Web Data Extraction, Personalization and Reasoning for the Semantic Web. In: GÓMEZ-PÉREZ, A.; EUZENAT, J. (Eds.). **The Semantic Web: Research and Applications: Second European Semantic Web Conference, ESWC 2005, Heraklion, Crete, Greece, May 29–June 1, 2005. Proceedings.** Lecture Notes in Computer Science. Berlin / Heidelberg Springer, v.3532, 2005. p.515-530.

BECHHOFER, S.; STEVENS, R.; LORD, P. **Ontology Driven Dynamic Linking of Biology Resources**. In: Pacific Symposium on Biocomputing 10. Hawaii. January, 2005. Available at: http://helix-web.stanford.edu/psb05/. Acessed: 2006-03-16.

BEN-ASHER, Y.; BERKOVSKY, S. **UNSO: Unspecified Ontologies for Peer-to-Peer E-Commerce Applications**. In: International Conference on Informatics, ICI. Cesme, Turkey. September, 2004. Available at: http://cslx.haifa.ac.il/~slavax/pubs.html. Acessed: 2006-04-05.

BENEVENTANO, D.; BERGAMASCHI, S. **The MOMIS methodology for integrating heterogeneous data sources**. In: Building the Information Society, IFIP 18th World Computer Congress, Topical Sessions. Toulouse, France: Kluwer. 22-27/08/2004, 2004. Available at: http://dblp.uni-trier.de/rec/bibtex/conf/ifip/BeneventanoB04. Acessed: 2006-03-16.

BENJAMINS, V. R. *et al.* **Six challenges for the Semantic Web**. In: KR2002 Workshop on Semantic Web. April, 2002. Available at: http://www.cs.man.ac.uk/~ocorcho/documents/KRR2002WS_BenjaminsEtAl.pdf. Acessed: 2005-08-22.

BERGAMASCHI, S. *et al.* Building a Tourism Information Provider with the Momis System. **Information Technology & Tourism Journal**, v.7, n.3-4, April, 30th. 2005. Available at: http://itt.ec3.at/volume2.asp?volume=7&number=3-4. Acessed: 2006-03-28.

BERNERS-LEE, T. **Semantic Web Road map.** 1998. Web Page. Available at: http://www.w3.org/DesignIssues/Semantic.html. Acessed: 2002-09-05.

______. **Semantic Web - XML2000.** 2000. Web Page. Available at: http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html. Acessed: 2005-08-22.

______. **Web for real people - WWW 2005.** 2005. Web Page. Available at: http://www.w3.org/2005/Talks/0511-keynote-tbl/#[17]. Acessed: 2006-11-10.

BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. The Semantic Web. **Scientific American**, May. 2001. Available at: http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21&ref=sciam. Acessed: 2005-08-22.

BLOOD, R. (Ed.) **We've Got Blog: How Weblogs Are Changing Our Culture.** 1st ed.: Perseus Books Group, 2002. 176 p.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **The Unified Modeling Language User Guide** 1st ed. New York, NY, USA: Addison-Wesley Professional, 1998. 482 p.

BORST, W. N. **Construction of Engineering Ontologies for Knowledge Sharing and Reuse**. 1997. 227 p. Thesis (PhD). Centre for Telematica and Information Technology, University of Twente, Enschede.

BREITMAN, K.; CASANOVA, M. A. **Semantic Web: Concepts, Technologies and Applications.** 1st ed.: Springer, 2006. 330 p.

BRICKLEY, D.; MILLER, L. **FOAF Vocabulary Specification - Namespace Document 27 July 2005 - ('Pages about Things' Edition).** Internet: xmlns.com, 2005. Web Page. Available at: http://xmlns.com/foaf/0.1/. Acessed: 2006-05-02.

BROEKSTRA, J. *et al.* **A Metadata Model for Semantics-Based Peer-to-Peer Systems**. In: WWW'03 Workshop on Semantics in Peer-to-Peer and Grid Computing - SemPGRID. Budapest, Hungary. May, 2003. Available at: http://www.cs.vu.nl/~frankh/abstracts/P2P@WWW03.html. Acessed: 2006-04-17.

BROEKSTRA, J.; KAMPMAN, A.; VAN HARMELEN, F. Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In: HORROCKS, I.; HENDLER, J. (Eds.). **The Semantic Web - ISWC 2002: First International Semantic Web Conference, Sardinia, Italy, June 9-12, 2002. Proceedings.** Lecture Notes in Computer Science. Berlin / Heidelberg: Springer, v.2342, 2002. p.54.

BRUIJN, J. D. **D16 The WSML Specification - WSML Working Draft 3 February 2005.** DERI, February, 2005, p. 2. Report. Available at: http://www.wsmo.org/TR/d16/v0.2/20050203/ Acessed: 2006-04-20.

CARR, L. *et al.* **Conceptual Linking: Ontology-based Open Hypermedia**. In: The Tenth International World Wide Web Conference (WWW10). Hong Kong. May, 2001. Available at: http://www10.org/cdrom/papers/246/index.html. Acessed: 2006-04-17.

______. **The Distributed Link Service: A Tool for Publishers, Authors and Readers**. In: Fourth International World Wide Web Conference. Boston, MA, USA. November, 1995. Available at: http://www.w3.org/Conferences/WWW4/Papers/178/. Acessed: 2006-04-17.

CARROLL, J. J. *et al.* **Jena: implementing the semantic web recommendations**. In: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters. New York, NY, USA ACM Press, 2004. Available at: http://doi.acm.org/10.1145/1013367.1013381. Acessed: 2006-10-18.

CATTELL, R. G. G.; BARRY, D. K. (Eds.). **The Object Data Standard: ODMG 3.0.** The Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, 2000.

CONALLEN, J. Modeling Web application architectures with UML **Communications of the ACM**, v.42, n.10, p.63-70, October. 1999. Available at: http://doi.acm.org/10.1145/317665.317677. Acessed: 2006-04-26.

CONTRERAS, J. *et al.* A Semantic Portal for the International Affairs Sector. In: MOTTA, E. *et al* (Eds.). **Engineering Knowledge in the Age of the SemanticWeb: 14th International Conference, EKAW 2004 Whittlebury Hall, UK, October 5-8, 2004. Proceedings.** Lecture Notes in Computer Science. Berlin / Heidelberg Springer, v.3257, 2004. p.203-215.

DING, L. *et al.* **Swoogle: a search and metadata engine for the semantic web** In: Proceedings of the thirteenth ACM international conference on Information and knowledge management. Washington, D.C., USA: ACM Press, 2004. Available at: http://doi.acm.org/10.1145/1031171.1031289 Acessed: 2006-04-14.

DSTC; IBM; SANDPIPER. **Ontology Definition Metamodel: Second Revised Submission to OMG/ RFP ad/2003-03-40, OMG Document: ad/05-04-13.** Internet: Object Management Group, 2005. Web Page. Available at: http://www.omg.org/cgi-bin/doc?ad/05-04-13. Acessed: 2005-08-21.

DUMBILL, E. **Describe open source projects with XML, Part 1: Keep project information up-to-date with the DOAP vocabulary.** IBM, February, 2004a. Report. Available at: http://www-128.ibm.com/developerworks/xml/library/x-osproj.html. Acessed: 2006-05-04.

______. **Describe open source projects with XML, Part 2: Keep project information up-to-date with the DOAP vocabulary.** IBM, March, 2004b. Report. Available at: http://www-128.ibm.com/developerworks/xml/library/x-osproj2/. Acessed: 2006-05-04.

______. **Describe open source projects with XML, Part 3: A first draft of the DOAP vocabulary.** IBM, June, 2004c. Report. Available at: http://www-128.ibm.com/developerworks/xml/library/x-osproj3/. Acessed: 2006-05-04.

EHRIG, M. *et al.* The SWAP Data and Metadata Model for Semantics-Based Peer-to-Peer Systems. In: SCHILLO, M. *et al* (Eds.). **Multiagent System Technologies.** Lecture Notes in Computer Science. Berlin / Heidelberg Springer, v.2831, 2003. p.144-155.

ELLIS, C. A.; GIBBS, S. J.; REIN, G. Groupware: some issues and experiences. **Communications of the ACM**, v.34, n.1, p.39-58, January. 1991. Available at: http://doi.acm.org/10.1145/99977.99987. Acessed: 2006-11-16.

FENSEL, D. *et al.* Introduction. In: FENSEL, D. *et al* (Eds.). **Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential.** The MIT Press, 2002. p.1-25.

______. Introduction. In: FENSEL, D. *et al* (Eds.). **Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential.** Cambridge, MA, USA: The MIT Press, 2003. p.1-25.

FILETO, R. **A abordagem POESIA para a integração de dados e serviços na Web semântica**. 2003. 150 p. Tese (Doutorado). Instituto de Computação, Universidade Estadual de Campinas (Unicamp), Campinas, SP, Brazil.

FLUIT, C.; SABOU, M.; VAN HARMELEN, F. Ontology-based Information Visualisation: Towards Semantic Web Applications. In: GEROIMENKO, V. (Ed.) **Visualising the Semantic Web.** 2 nd ed. Berlin: Springer Verlag, 2005.

FROST, R. A. **Introduction to knowledge base systems.** New York, NY, USA: Macmillan Publishing Company, 1986. 677 p.

GAMMA, E. *et al.* **Design Patterns: Elements of Reusable Object-Oriented Software.** 1st ed. New York, NY, USA: Addison-Wesley Professional, 1995. 395 p.

GAO, Z. *et al.* DynamicView: Distribution, Evolution and Visualization of Research Areas in Computer Science. In: GIL, Y. *et al* (Eds.). **The Semantic Web – ISWC 2005: 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland,**

November 6-10, 2005. Proceedings. Lecture Notes in Computer Science. Berlin / Heidelberg Springer, v.3729, 2005. p.1054-1058.

GREIF, I. (Ed.) **Computer-supported cooperative work: a book of readings.** San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988. 781 p.

GRØNBÆK, K.; SLOTH, L.; ØRBÆK, P. **Webvise: Browser and Proxy Support for Open Hypermedia Structuring Mechanisms on the WWW**. In: The Eighth International World Wide Web Conference (WWW10). Toronto, ON, Canada: Elsevier Science. May, 1999. Available at: http://www8.org/w8-papers/3a-search-query/webvise/webvise.html. Acessed: 2006-04-18.

GRUBER, T. R. A translation approach to portable ontology specifications. **Knowledge Acquisition**, v.5, n.2, p.199-220. 1993. Available at: http://www.sciencedirect.com/science/article/B6WMS-45P67XR-D/2/b9a5cb273fc6497ae1b4fefad29b30b5. Acessed: 2005-08-22.

GUARINO, N. Formal ontology, conceptual analysis and knowledge representation. **International Journal of Human-Computer Studies**, v.43, n.5-6, p.625-640. 1995. Available at: http://dx.doi.org/10.1006/ijhc.1995.1066. Acessed: 2006-07-18.

______. Understanding, building and using ontologies. **International Journal of Human-Computer Studies**, v.46, n.2-3, p.293-310. 1997. Available at: http://dx.doi.org/10.1006/ijhc.1996.0091. Acessed: 2006-07-18.

______. **Formal Ontology in Information Systems**. In: Proceedings of FOIS'98 - Formal Ontology in Information Systems. Trento, Italy: IOS Press, 1998. Available at: http://www.loa-cnr.it/Publications.html#Pub1998. Acessed: 2005-10-20.

GUARINO, N.; GIARETTA, P. Ontologies and Knowledge Bases: Towards a Terminological Clarification. In: MARS, N. (Ed.) **Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing.** Amsterdam, NL: IOS Press, 1995. p.25-32.

HAASE, P. *et al*. Bibster - A Semantics-Based Bibliographic Peer-to-Peer System. In: MCILRAITH, S. A.; PLEXOUSAKIS, D.; HARMELEN, F. V. (Eds.). **The Semantic Web – ISWC 2004: Third International Semantic Web Conference, November 7-11, 2004. Proceedings.** Lecture Notes in Computer Science. Berlin / Heidelberg: Springer, v.3298, 2004. p.122-136.

HARTH, A. SECO: mediation services for semantic Web data. **Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]**, v.19, n.3, p.66-71. 2004. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=29151&arnumber=1315543&count=15&index=9. Acessed: 2006-03-02.

HARTMANN, J.; SURE, Y. An infrastructure for scalable, reliable semantic portals. **Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]**, v.19, n.3, p.58-65. 2004. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1315542. Acessed: 2006-03-02.

HEFLIN, J.; HENDLER, J.; LUKE, S. SHOE: A Blueprint for the Semantic Web. In: FENSEL, D. *et al* (Eds.). **Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential.** Cambridge, MA, USA: The MIT Press, 2003. p.29-63.

HEFLIN, J.; HENDLER, J. A. **Dynamic Ontologies on the Web**. In: Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence. AAAI Press / The MIT Press, 2000. Available at: http://portal.acm.org/citation.cfm?id=647288.721581&coll=GUIDE&dl=GUIDE&CFID=52921068&CFTOKEN=44875420. Acessed: 2005-08-22.

HENDLER, J. Agents and the Semantic Web. **Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]**, v.16, n.2, p.30-37. 2001. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=19905&arnumber=920597&count=14&index=6. Acessed: 2006-03-02.

HENZE, N.; HERRLICH, M. **The Personal Reader: A Framework for Enabling Personalization Services on the Semantic Web**. In: Twelfth GI - Workshop on Adaptation and User Modeling in interactive Systems (ABIS 04). Berlin, Germany. October, 2004. Available at: http://idefix.pms.ifi.lmu.de:8080/rewerse/index.html#REWERSE-RP-2004-58. Acessed: 2006-04-19.

HENZE, N.; KRIESELL, M. **Personalization functionality for the semantic web: architectural outline and first sample implementation**. In: First International Workshop on Engineering the Adaptive Web in conjunction with Third International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2004). Eindhoven, The Netherlands. June, 2004. Available at: http://idefix.pms.ifi.lmu.de:8080/rewerse/index.html#REWERSE-RP-2004-31. Acessed: 2006-04-19.

HORROCKS, I. *et al.* **Semantic Web Architecture: Stack or Two Towers?** . In: Third Workshop on Principles and Practices of Semantic Web Reasoning. Dagstuhl, Germany: Springer Verlag, 2005. Available at: http://www-db.research.bell-labs.com/user/pfps/publications/. Acessed: 2005-10-14.

HÜBNER, S. *et al.* Ontology-based search for interactive digital maps. **Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]**, v.19, n.3, p.80-86. 2004. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=29151&arnumber=1315545&count=15&index=11. Acessed: 2006-03-03.

HYVÖNEN, E. *et al.* MuseumFinland--Finnish museums on the semantic web. **Web Semantics: Science, Services and Agents on the World Wide Web**, v.3, n.2-3, p.224-241. 2005. Available at: http://www.sciencedirect.com/science/article/B758F-4GXVGBB-1/2/31b4fa1159b5bb40335853c6c25ecd50 Acessed: 2006-03-09.

INRIA. **Knowledge Management Platform (KmP) - A Semantic Web Service for the Cartography of Competences in the Telecom Valley of Sophia Antipolis** Internet: ACACIA, 2006. Web Page. Available at: http://www-sop.inria.fr/acacia/soft/kmp.html. Acessed: 2006-04-13.

JASPER, R.; USCHOLD, M. **A Framework for Understanding and Classifying Ontology Applications**. In: KAW'99 Twelfth Workshop on Knowledge Acquisition, Modeling and Management. Banff, Alberta, Canada, 1999. Available at: http://sern.ucalgary.ca/KSI/KAW/KAW99/papers/Uschold2/final-ont-apn-fmk.pdf. Acessed: 2005-08-23.

KAHAN, J. *et al.* **Annotea: An Open RDF Infrastructure for Shared Web Annotations**. In: The Tenth International World Wide Web Conference (WWW10). Hong Kong. May, 2001. Available at: http://www10.org/cdrom/papers/488/index.html. Acessed: 2006-04-17.

KELLER, R. M. *et al.* SemanticOrganizer: A Customizable Semantic Repository for Distributed NASA Project Teams. In: MCILRAITH, S. A.; PLEXOUSAKIS, D.; HARMELEN, F. V. (Eds.). **The Semantic Web - ISWC 2004 - Third International Semantic Web Conference, Hiroshima, Japan, November 7-11, 2004. Proceedings.** Lecture Notes in Computer Science. Springer Berlin / Heidelberg, v.3298, 2004. p.767-781.

KERRIGAN, M. **D9.1v0.2 Web Service Modeling Toolkit (WSMT) - WSMX Working Draft 25 April 2005.** DERI, April, 2005, p. 9. Report. Available at: http://www.wsmo.org/TR/d9/d9.1/v0.2/20050425 Acessed: 2006-04-20.

KLEIN, M.; VISSER, U. Guest Editors' Introduction: Semantic Web Challenge 2003. **Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]**, v.19, n.3, p.31-33. 2004. Available at: http://ieeexplore.ieee.org/iel5/9670/29151/01315538.pdf?tp=&arnumber=1315538&isnumber=29151. Acessed: 2005-08-22.

______. Semantic Web Challenge 2004. **Web Semantics: Science, Services and Agents on the World Wide Web**, v.3, n.2-3, p.209-210. 2005. Available at: http://www.sciencedirect.com/science/article/B758F-4GMGWFM-1/2/41789ecb344048bcec05c49164326823 Acessed: 2006-03-07.

KOIVUNEN, M.-R. **Annotea and Semantic Web Supported Collaboration** In: UserSWeb: Workshop on End User Aspects of the Semantic Web at ESWC05. Heraklion, Crete, Greece: CEUR Workshop Proceedings (CEUR-WS.org) May, 2005. Available at: http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS//Vol-137/. Acessed: 2006-03-16.

KRUCHTEN, P. B. The 4+1 View Model of architecture. **Software, IEEE**, v.12, n.6, p.42-50. 1995. Available at: http://ieeexplore.ieee.org/search/wrapper.jsp?arnumber=469759. Acessed: 2006-11-15.

LÉGER, A. *et al.* **D2.2 Successful Scenarios for Ontology-based Applications V1.0.** 31/05/2002, 2002, p. 100. Available at: http://ontoweb.org/Members/huro/MyPublications/OntoWeb%20Deliverable%202.2/view. Acessed: 2005-09-29.

LEUF, B.; CUNNINGHAM, W. **The Wiki Way: Collaboration and Sharing on the Internet.** New York, NY, USA: Addison-Wesley Professional, 2001. 440 p.

LI, D. *et al.* Search on the Semantic Web. **Computer**, v.38, n.10, p.62-69. 2005. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=32474&arnumber=1516060&count=18&index=8. Acessed: 2006-03-15.

LIMA, F. **Modelagem semântica de aplicações na WWW**. 2003. 128 p. Tese (Doutorado). Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), Rio de Janeiro, RJ, Brazil.

MAEDCHE, A. *et al.* SEAL - Tying Up Information Integration and Web Site Management by Ontologies. **Data Engineering Bulletin**, v.25, n.1, p.10-17. 2002. Available at: http://www.informatik.uni-trier.de/~ley/db/journals/debu/debu25.html. Acessed: 2006-04-06.

MÄKELÄ , E. *et al.* OntoViews - A Tool for Creating Semantic Web Portals. In: MCILRAITH, S. A.; PLEXOUSAKIS, D.; VAN HARMELEN, F. (Eds.). **The Semantic Web - ISWC 2004 - Third International Semantic Web Conference, Hiroshima, Japan, November 7-11, 2004. Proceedings.** Lecture Notes in Computer Science. Berlin / Heidelberg Springer, v.3298, 2004. p.797-811.

MCGUINNESS, D. L. Ontologies Come of Age. In: FENSEL, D. *et al* (Eds.). **Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential.** Cambridge, MA, USA: The MIT Press, 2003. p.171-194.

MICHALOWSKI, M. *et al.* Retrieving and semantically integrating heterogeneous data from the Web. **Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]**, v.19, n.3, p.72-79. 2004. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=29151&arnumber=1315544&count=15&index=10. Acessed: 2006-03-03.

MIKA, P. Flink: Semantic Web technology for the extraction and analysis of social networks. **Web Semantics: Science, Services and Agents on the World Wide Web**, v.3, n.2-3, p.211-223. 2005a. Available at: http://www.sciencedirect.com/science/article/B758F-4GP1VK5-1/2/ad4ed5e753baf8818022ee4ec72e3a83 Acessed: 2006-03-15.

______. **User Guide for Elmo.** Internet: openRDF.org, 2005b. Web page. Available at: http://www.openrdf.org/doc/elmo/users/userguide.html. Acessed: 2006-10-28.

MORAN, M.; ZAREMBA, M.; MOCAN, A. **D13.8 v0.1 Integration of WSMX with other SWS Systems - WSMX Final Draft 14-06-2005.** DERI, June, 2005, p. 15. Report. Available at: http://www.wsmo.org/TR/d13/d13.8/v0.1/20050614/. Acessed: 2006-04-18.

NOWACK, B. CONFOTO: A Semantic Browsing and Annotation Service for Conference Photos. In: GIL, Y. *et al* (Eds.). **The Semantic Web – ISWC 2005: 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland, November 6-10, 2005. Proceedings.** Lecture Notes in Computer Science. Berlin / Heidelberg Springer, v.3729, 2005. p.1067-1070.

NUMA, K. *et al.* **Egocentric Search Based on RSS**. In: Poster track at the 3rd International Semantic Web Conference (ISWC2004). Hiroshima, Japan. November,

2004. Available at: http://iswc2004.semanticweb.org/posters/index.html. Acessed: 2006-09-03.

OHMUKAI, I.; TAKEDA, H.; NUMA, K. **Personal Knowledge Publishing Suite with Weblog**. In: WWW 2004 Workshop on the Weblogging Ecosystem: Aggregation, Analysis and Dynamics. 2004. Available at: http://www-kasm.nii.ac.jp/~takeda/research/projects.html. Acessed: 2006-09-18.

ØSTERBYE, K.; WIIL, U. K. **The flag taxonomy of open hypermedia systems**. In: The seventh ACM conference on Hypertext. Bethesda, Maryland, United States: ACM Press, 1996. Available at: http://doi.acm.org/10.1145/234828.234841. Acessed: 2006-04-18.

PALMA, R.; HAASE, P. Oyster - Sharing and Re-using Ontologies in a Peer-to-Peer Community. In: GIL, Y. *et al* (Eds.). **The Semantic Web – ISWC 2005: 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland, November 6-10, 2005. Proceedings.** Lecture Notes in Computer Science. Berlin / Heidelberg Springer, v.3729, 2005. p.1059-1062.

PATEL-SCHNEIDER, P. F. A Revised Architecture for Semantic Web Reasoning. In: BRY, F.; OHLBACH, H. J. (Eds.). **Principles and Practice of Semantic Web Reasoning.** Lecture Notes in Computer Science. Springer, v.3703, 2005. p.32-36.

PLAISANT, C. *et al*. Interface and data architecture for query preview in networked information systems. **ACM Transactions on Information Systems (TOIS)**, v.17, n.3, p.320-341 1999. Available at: http://doi.acm.org/10.1145/314516.314522 Acessed: 2006-04-10.

RAMAGEM, D. B.; MARGERIN, B.; KENDALL, J. AnnoTerra: building an integrated earth science resource using semantic Web technologies. **Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]**, v.19, n.3, p.48-57. 2004. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1315541. Acessed: 2006-03-02.

RASKIN, R. **Enabling Semantic Interoperability for Earth Science Data.** NASA, p. 7. Final Report to NASA Earth Science Technology Office (ESTO). Available at: http://sweet.jpl.nasa.gov/products.html. Acessed: 2006-04-04.

RASKIN, R.; PAN, M. **Semantic Web for Earth and Environmental Terminology**. In: Workshop on Semantic Web Technologies for Searching and Retrieving Scientific Data at ISWC2003. Sanibel Island, Florida, USA: CEUR Workshop Proceedings (CEUR-WS.org). October, 2003. Available at: http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-83/. Acessed: 2006-11-21.

ROMAN, D.; LAUSEN, H.; KELLER, U. **D2v02. Web Service Modeling Ontology - Standard (WSMO - Standard) - WSMO Working Draft 06 March 2004.** DERI, March, 2004, p. 19. Report. Available at: http://www.wsmo.org/2004/d2/v0.2/20040306/. Acessed: 2006-04-20.

ROSSI, G.; SCHWABE, D.; LYARDET , F. Web Application Models Are More Than Conceptual Models. In: **Proceedings of the Workshops on Evolution and Change in Data Management, Reverse Engineering in Information Systems, and the World**

Wide Web and Conceptual Modeling. Lecture Notes In Computer Science. Springer-Verlag, v.1727, 1999. p.239 - 253.

SCHLOSSER, M. *et al.* **A scalable and ontology-based P2P infrastructure for Semantic Web Services**. In: Second International Conference on Peer-to-Peer Computing (P2P 2002). September, 2002. Available at: http://ieeexplore.ieee.org/iel5/8107/22420/01046319.pdf?tp=&arnumber=1046319&isnumber=22420. Acessed: 2006-04-20.

SEEMAN, L. **The semantic web, web accessibility, and device independence**. In: Proceedings of the 2004 international cross-disciplinary workshop on Web accessibility (W4A). New York, NY, USA: ACM Press, 2004. Available at: http://www.w4a.info/2004/about/pub.shtml
http://doi.acm.org/10.1145/990657.990669 Acessed: 2006-03-07.

SHABAN-NEJAD, A. *et al.* The FungalWeb Ontology: Semantic Web Challenges in Bioinformatics and Genomics. In: GIL, Y. *et al* (Eds.). **The Semantic Web – ISWC 2005: 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland, November 6-10, 2005. Proceedings.** Lecture Notes in Computer Science. Berlin / Heidelberg: Springer, v.3729, 2005. p.1063-1066.

______. **The FungalWeb Ontology: The Core of a Semantic Web Application for Fungal Genomics**. In: The 1ST Canadian Semantic Web Interest Group Meeting (SWIG'04). Montreal, Quebec, Canada. November, 2004. Available at: http://www.cs.concordia.ca/~haarslev/publications/. Acessed: 2006-04-19.

SHADBOLT, N. *et al.* CS AKTive Space, or how we learned to stop worrying and love the semantic Web. **Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]**, v.19, n.3, p.41-47. 2004. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=29151&arnumber=1315540&count=15&index=6. Acessed: 2006-03-03.

SILVA, V. *et al.* Taming Agents and Objects in Software Engineering. In: GARCIA, A. *et al* (Eds.). **Software Engineering for Large-Scale Multi-Agent Systems: Research Issues and Practical Applications.** Lecture Notes in Computer Science. Springer-Verlag GmbH, v. 2603, 2003. p.1-26.

SOMMERVILLE, I. **Software Engineering.** 6th ed. New York, NY, USA: Addison Wesley, 2000.

STEGERS, R.; FEKKES, P.; STUCKENSCHMIDT, H. MusiDB: A personalized search engine for music. **Web Semantics: Science, Services and Agents on the World Wide Web**, v.(In Press, Corrected Proof). 2006. Available at: http://www.sciencedirect.com/science/article/B758F-4H9YC8X-2/2/1b7f0a35ee6804ad726a5a327cb67696 Acessed: 2006-03-07.

STUCKENSCHMIDT, H. *et al.* Exploring large document repositories with RDF technology: the DOPE project. **Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]**, v.19, n.3, p.34-40. 2004. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=29151&arnumber=1315539&count=15&index=5. Acessed: 2006-03-02.

SWAP EU IST-2001-34103 Final Report. Institute AIFB, University of Karlsruhe, November, 2004, p. 88. Report. Available at: http://swap.semanticweb.org/public/deliverables.htm. Acessed: 2006-04-19.

SWARTZ, A. MusicBrainz: a semantic Web service. **Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]**, v.17, n.1, p.76-77. 2002. Available at: http://ieeexplore.ieee.org/search/wrapper.jsp?arnumber=988466. Acessed: 2006-04-12.

SZYPERSKI, C. **Component software: beyond object-oriented programming.** New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1998. 411 p.

TAKEDA, H.; OHMUKAI, I. **Semblog Project**. In: Activities on Semantic Web Technologies in Japan - a WWW2005 Workshop. Japan. May, 2005. Available at: http://www-kasm.nii.ac.jp/papers/ai-lab/abstract/select-bibE.cgi?title=KasM+Papers&option=&pat11=Takeda. Acessed: 2006-03-03.

TAZZOLI, R.; CASTAGNA, P.; CAMPANINI, S. E. **Towards a Semantic Wiki Wiki Web**. In: Poster track at the 3rd International Semantic Web Conference (ISWC2004). Hiroshima, Japan. November, 2004. Available at: http://iswc2004.semanticweb.org/posters/index.html. Acessed: 2006-03-09.

TEJADA, S.; KNOBLOCK, C. A.; MINTON , S. **Learning domain-independent string transformation weights for high accuracy object identification**. In: Eighth ACM SIGKDD international conference on Knowledge discovery and data mining. Edmonton, Alberta, Canada: ACM Press, 2002. Available at: http://doi.acm.org/10.1145/775047.775099 Acessed: 2006-04-20.

THAKKAR, S.; AMBITE, J. L.; KNOBLOCK, C. A. **A view integration approach to dynamic composition of web services**. In: International Conference on Automated Planning and Scheduling - Workshop on Planning for Web Services. Trento, Italy: AAAI Press, 2003. Available at: http://www.isi.edu/cgi-bin/ariadne/integration/papers.cgi. Acessed: 2006-04-20.

TUMMARELLO, G.; MORBIDONI, C. **The DBin Semantic Web Platform: An Overview.** Internet: DBin Project, 2005. Slides. Available at: http://www.dbin.org/papers.php. Acessed: 2006-04-04.

TUMMARELLO, G. *et al.* **The DBin Semantic Web Platform: An Overview**. In: The Semantic Computing Initiative (SeC 2005) - From Semantic Web to Semantic World - a WWW2005 Workshop. Japan. May, 2005. Available at: http://www.instsec.org/2005ws/papers/tummarello.pdf. Acessed: 2006-03-07.

USCHOLD, M. Where are the semantics in the semantic web? **AI Magazine**, v.24, n.3, p.25-36, September. 2003. Available at: http://portal.acm.org/citation.cfm?id=958674&jmp=abstract&dl=GUIDE&dl=GUIDE&CFID=311776&CFTOKEN=33586894#abstract. Acessed: 2006-07-27.

VAN HEIJST, G.; SCHREIBER, A. T.; WIELINGA, B. J. Roles are not classes: a reply to Nicola Guarino. **International Journal of Human-Computer Studies**, v.46, n.2-3, p.311-318. 1997. Available at: http://dx.doi.org/10.1006/ijhc.1996.0092. Acessed: 2006-07-18.

VISSER, U.; KLEIN, M. **Semantic web Challenge: A new application award.** 2005. Flyer. Available at: http://www.informatik.uni-bremen.de/swc/swcflyer05.pdf. Acessed: 2005-11-09.

W3C. **DAML+OIL (March 2001) Reference Description - W3C Note 18 December 2001.** Internet: World Wide Web Consortium, 2001. Web Page. Available at: http://www.w3.org/TR/daml+oil-reference. Acessed: 2005-10-11.

______. **OWL Web Ontology Language Guide - W3C Recommendation 10 February 2004.** Internet: World Wide Web Consortium, 2004a. Web Page. Available at: http://www.w3.org/TR/2004/REC-owl-guide-20040210. Acessed: 2005-11-07.

______. **OWL Web Ontology Language Overview: W3C Recommendation 10 February 2004.** Internet: World Wide Web Consortium, 2004b. Web Page. Available at: http://www.w3.org/TR/2004/REC-owl-features-20040210/. Acessed: 2005-08-23.

______. **OWL Web Ontology Language Semantics and Abstract Syntax: W3C Recommendation 10 February 2004** Internet: World Wide Web Consortium, 2004c. Web Page. Available at: http://www.w3.org/TR/2004/REC-owl-semantics-20040210/. Acessed: 07/11/2005.

______. **OWL Web Ontology Language Use Cases and Requirements: W3C Recommendation 10 February 2004.** Internet: World Wide Web Consortium, 2004d. Web Page. Available at: http://www.w3.org/TR/2004/REC-webont-req-20040210/. Acessed: 18/07/2005.

______. **RDF (Resource Description Framework) Primer: W3C Recommendation 10 February 2004.** Internet: World Wide Web Consortium, 2004e. Web Page. Available at: http://www.w3.org/TR/2004/REC-rdf-primer-20040210/. Acessed: 2005-08-22.

______. **RDF Vocabulary Description Language 1.0: RDF Schema - W3C Recommendation 10 February 2004.** Internet: World Wide Web Consortium, 2004f. Web Page. Available at: http://www.w3.org/TR/2004/REC-rdf-schema-20040210/. Acessed: 2006-06-13.

______. **World Wide Web Consortium Issues RDF and OWL Recommendations: Semantic Web emerges as commercial-grade infrastructure for sharing data on the Web.** Internet: World Wide Web Consortium, 2004g. Web Page. Available at: http://www.w3.org/2004/01/sws-pressrelease.html.en. Acessed: 2005-11-07.

______. **Ontology Driven Architectures and Potential Uses of the Semantic Web in Software Engineering Editors' Draft $Date: 2005/06/30 13:34:22 $.** Internet: World Wide Web Consortium, 2005a. Web Page. Available at: http://www.w3.org/2001/sw/BestPractices/SE/ODA/. Acessed: 2005-08-11.

______. **Semantic Web Activity Statement.** Internet: World Wide Web Consortium 2005b. Web Page. Available at: http://www.w3.org/2001/sw/Activity. Acessed: 2005-05-24.

______. **Semantic Web Best Practices and Deployment Minutes from Face to Face Meeting.** Internet: World Wide Web Consortium 2005c. Web Page. Available at: http://www.w3.org/2005/03/03-swbp-minutes. Acessed: 2005-11-29.

ZAREMBA, M.; MORAN, M. **D13.4v0.2 WSMX Architecture - WSMX Working Draft 21-01-2005.** DERI, January, 2005. Report. Available at: http://www.wsmo.org/2005/d13/d13.4/v0.2/20050121/ Acessed: 2006-03-28.

ZYL, J. V.; CORBETT, DAN. **A Framework for Comparing the use of a Linguistic Ontology in an Application** In: ECAI'00 Workshop on Applications of Ontologies and Problem-Solving Methods. Berlin, 2000a. Available at: http://citeseer.ist.psu.edu/443031.html. Acessed: 2005-10-13.

______. **Population of a Framework for Understanding and Classifying Ontology Applications** In: ECAI'00 Workshop on Applications of Ontologies and Problem-Solving Methods. Berlin, 2000b. Available at: http://citeseer.ist.psu.edu/442857.html. Acessed: 2005-10-13.

Annex A - The DOAP Vocabulary

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:vs="http://www.w3.org/2003/06/sw-vocab-status/ns#"
  xmlns:foaf="http://xmlns.com/foaf/0.1/"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:doap="http://usefulinc.com/ns/doap#"
>

<!-- Copyright © 2004 Edd Dumbill <edd@usefulinc.com>
       $LastChangedDate: 2005-11-05 12:40:58 +0000 (Sat, 05 Nov 2005) $
       $LastChangedRevision: 245 $
       $LastChangedBy: edd $
-->

<!-- about this schema -->

<owl:Ontology rdf:about="http://usefulinc.com/ns/doap#">
       <owl:imports rdf:resource="http://xmlns.com/foaf/0.1/index.rdf" />
       <dc:title>Description of a Project (DOAP) vocabulary</dc:title>
       <dc:description>The Description of a Project (DOAP) vocabulary,
described using W3C RDF Schema and the Web Ontology Language.</dc:description>
       <dc:description xml:lang="fr">Le vocabulaire Description Of A Project
(DOAP, Description D'Un Projet), décrit en utilisant RDF Schema du W3C et
OWL.</dc:description>
       <dc:description xml:lang="es">El vocabulario Description of a Project
(DOAP, Descripción de un Proyecto), descrito usando RDF Schema de W3C y Web
Ontology Language.</dc:description>
       <dc:creator>Edd Dumbill</dc:creator>
       <dc:format>application/rdf+xml</dc:format>
       <dc:rights>Copyright © 2004 Edd Dumbill</dc:rights>
       <foaf:maker>
              <foaf:Person>
                     <foaf:name>Edd Dumbill</foaf:name>
                     <foaf:mbox rdf:resource="mailto:edd@usefulinc.com" />
              </foaf:Person>
       </foaf:maker>
</owl:Ontology>

<!-- Classes are listed first -->

<rdfs:Class rdf:about="http://usefulinc.com/ns/doap#Project">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

<rdfs:label xml:lang="en">Project</rdfs:label>
       <rdfs:label xml:lang="fr">Projet</rdfs:label>
       <rdfs:label xml:lang="es">Proyecto</rdfs:label>

       <rdfs:comment xml:lang="en">A project.</rdfs:comment>
       <rdfs:comment xml:lang="fr">Un projet.</rdfs:comment>
       <rdfs:comment xml:lang="es">Un proyecto.</rdfs:comment>

       <rdfs:subClassOf rdf:resource="http://xmlns.com/wordnet/1.6/Project" />
       <rdfs:subClassOf rdf:resource="http://xmlns.com/foaf/0.1/Project" />
</rdfs:Class>

<rdfs:Class rdf:about="http://usefulinc.com/ns/doap#Version">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

<rdfs:label xml:lang="en">Version</rdfs:label>
       <rdfs:label xml:lang="fr">Version</rdfs:label>
       <rdfs:label xml:lang="es">Versión</rdfs:label>

       <rdfs:comment xml:lang="en">Version information of a project
release.</rdfs:comment>
       <rdfs:comment xml:lang="fr">Détails sur une version d'une realease d'un
projet.</rdfs:comment>
       <rdfs:comment xml:lang="es">Información sobre la versión de un release
del proyecto.</rdfs:comment>
</rdfs:Class>
```

```
<rdfs:Class rdf:about="http://usefulinc.com/ns/doap#Repository">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">Repository</rdfs:label>
	<rdfs:label xml:lang="fr">Dépôt</rdfs:label>
	<rdfs:label xml:lang="es">Repositorio</rdfs:label>

	<rdfs:comment xml:lang="en">Source code repository.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Dépôt du code source.</rdfs:comment>
	<rdfs:comment xml:lang="es">Repositorio del código
fuente.</rdfs:comment>
</rdfs:Class>

<rdfs:Class rdf:about="http://usefulinc.com/ns/doap#SVNRepository">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">Subversion Repository</rdfs:label>
	<rdfs:label xml:lang="fr">Dépôt Subversion</rdfs:label>
	<rdfs:label xml:lang="es">Repositorio Subversion</rdfs:label>

	<rdfs:comment xml:lang="en">Subversion source code
repository.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Dépôt Subversion du code
source.</rdfs:comment>
	<rdfs:comment xml:lang="es">Repositorio Subversion del código
fuente.</rdfs:comment>

	<rdfs:subClassOf rdf:resource="http://usefulinc.com/ns/doap#Repository"
/>
</rdfs:Class>

<rdfs:Class rdf:about="http://usefulinc.com/ns/doap#BKRepository">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">BitKeeper Repository</rdfs:label>
	<rdfs:label xml:lang="fr">Dépôt BitKeeper</rdfs:label>
	<rdfs:label xml:lang="es">Repositorio BitKeeper</rdfs:label>

	<rdfs:comment xml:lang="en">BitKeeper source code
repository.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Dépôt BitKeeper du code
source.</rdfs:comment>
	<rdfs:comment xml:lang="es">Repositorio BitKeeper del código
fuente.</rdfs:comment>

	<rdfs:subClassOf rdf:resource="http://usefulinc.com/ns/doap#Repository"
/>
</rdfs:Class>

<rdfs:Class rdf:about="http://usefulinc.com/ns/doap#CVSRepository">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">CVS Repository</rdfs:label>
	<rdfs:label xml:lang="fr">Dépôt CVS</rdfs:label>
	<rdfs:label xml:lang="es">Repositorio CVS</rdfs:label>

	<rdfs:comment xml:lang="en">CVS source code repository.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Dépôt CVS du code source.</rdfs:comment>
	<rdfs:comment xml:lang="es">Repositorio CVS del código
fuente.</rdfs:comment>

	<rdfs:subClassOf rdf:resource="http://usefulinc.com/ns/doap#Repository"
/>
</rdfs:Class>

<rdfs:Class rdf:about="http://usefulinc.com/ns/doap#ArchRepository">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">GNU Arch repository</rdfs:label>
	<rdfs:label xml:lang="fr">Dépôt GNU Arch</rdfs:label>
	<rdfs:label xml:lang="es">Repositorio GNU Arch</rdfs:label>
```

```
       <rdfs:comment xml:lang="en">GNU Arch source code
repository.</rdfs:comment>
       <rdfs:comment xml:lang="fr">Dépôt GNU Arch du code
source.</rdfs:comment>
       <rdfs:comment xml:lang="es">Repositorio GNU Arch del código
fuente.</rdfs:comment>

       <rdfs:subClassOf rdf:resource="http://usefulinc.com/ns/doap#Repository"
/>
</rdfs:Class>

<!-- Properties -->

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#name">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

       <rdfs:label xml:lang="en">name</rdfs:label>
       <rdfs:label xml:lang="fr">nom</rdfs:label>
       <rdfs:label xml:lang="es">nombre</rdfs:label>

       <rdfs:comment xml:lang="en">A name of something.</rdfs:comment>
       <rdfs:comment xml:lang="fr">Le nom de quelque chose.</rdfs:comment>
       <rdfs:comment xml:lang="es">El nombre de algo.</rdfs:comment>

       <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"
/>

       <rdfs:subPropertyOf rdf:resource="http://www.w3.org/2000/01/rdf-
schema#label" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#homepage">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

       <rdfs:label xml:lang="en">homepage</rdfs:label>
       <rdfs:label xml:lang="fr">page web</rdfs:label>
       <rdfs:label xml:lang="es">página web</rdfs:label>

       <rdfs:comment xml:lang="en">URL of a project's homepage, associated with
exactly one project.</rdfs:comment>
       <rdfs:comment xml:lang="fr">L'URL de la page web d'un projet, associée
avec un unique projet.</rdfs:comment>
       <rdfs:comment xml:lang="es">El URL de la página de un proyecto, asociada
con exactamente un proyecto.</rdfs:comment>

       <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#InverseFunctionalProperty" />

       <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />

       <rdfs:subPropertyOf rdf:resource="http://xmlns.com/foaf/0.1/homepage" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#old-homepage">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

       <rdfs:label xml:lang="en">old homepage</rdfs:label>
       <rdfs:label xml:lang="fr">ancienne page web</rdfs:label>
       <rdfs:label xml:lang="es">página web antigua</rdfs:label>

       <rdfs:comment xml:lang="en">URL of a project's past homepage, associated
with exactly one project.</rdfs:comment>
       <rdfs:comment xml:lang="fr">L'URL d'une ancienne page web d'un projet,
associée avec un unique projet.</rdfs:comment>
       <rdfs:comment xml:lang="es">El URL de la antigua página de un proyecto,
asociada con exactamente un proyecto.</rdfs:comment>

       <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#InverseFunctionalProperty" />

       <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />

       <rdfs:subPropertyOf rdf:resource="http://xmlns.com/foaf/0.1/homepage" />
```

```
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#created">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">created</rdfs:label>
	<rdfs:label xml:lang="fr">créé</rdfs:label>
	<rdfs:label xml:lang="es">creado</rdfs:label>

	<rdfs:comment xml:lang="en">Date when something was created, in YYYY-MM-
DD form. e.g. 2004-04-05</rdfs:comment>
	<rdfs:comment xml:lang="fr">Date à laquelle a été créé quelque chose, au
format AAAA-MM-JJ (par ex. 2004-04-05)</rdfs:comment>
	<rdfs:comment xml:lang="es">Fecha en la que algo fue creado, en formato
AAAA-MM-DD. e.g. 2004-04-05</rdfs:comment>

	<rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"
/>
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#shortdesc">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">short description</rdfs:label>
	<rdfs:label xml:lang="fr">description courte</rdfs:label>
	<rdfs:label xml:lang="es">descripción corta</rdfs:label>

	<rdfs:comment xml:lang="en">Short (8 or 9 words) plain text description
of a project.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Texte descriptif concis (8 ou 9 mots) d'un
projet.</rdfs:comment>
	<rdfs:comment xml:lang="es">Descripción corta (8 o 9 palabras) en texto
plano de un proyecto.</rdfs:comment>

	<rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"
/>
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#description">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">description</rdfs:label>
	<rdfs:label xml:lang="fr">description</rdfs:label>
	<rdfs:label xml:lang="es">descripción</rdfs:label>

	<rdfs:comment xml:lang="en">Plain text description of a project, of 2-4
sentences in length.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Texte descriptif d'un projet, long de 2 à 4
phrases.</rdfs:comment>
	<rdfs:comment xml:lang="es">Descripción en texto plano de un proyecto,
de 2 a 4 enunciados de longitud.</rdfs:comment>

	<rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"
/>
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#release">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">release</rdfs:label>
	<rdfs:label xml:lang="fr">release</rdfs:label>
	<rdfs:label xml:lang="es">release</rdfs:label>

	<rdfs:comment xml:lang="en">A project release.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Une release (révision) d'un
projet.</rdfs:comment>
	<rdfs:comment xml:lang="es">Un release (versión) de un
proyecto.</rdfs:comment>

	<rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />

	<rdfs:range rdf:resource="http://usefulinc.com/ns/doap#Version" />
</rdf:Property>
```

```
<rdf:Property rdf:about="http://usefulinc.com/ns/doap#mailing-list">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

       <rdfs:label xml:lang="en">mailing list</rdfs:label>
       <rdfs:label xml:lang="fr">liste de diffusion</rdfs:label>
       <rdfs:label xml:lang="es">lista de correo</rdfs:label>

       <rdfs:comment xml:lang="en">Mailing list home page or email
address.</rdfs:comment>
       <rdfs:comment xml:lang="fr">Page web de la liste de diffusion, ou
adresse de courriel.</rdfs:comment>
       <rdfs:comment xml:lang="es">Página web de la lista de correo o dirección
de correo.</rdfs:comment>

       <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#category">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

       <rdfs:label xml:lang="en">category</rdfs:label>
       <rdfs:label xml:lang="fr">catégorie</rdfs:label>
       <rdfs:label xml:lang="es">categoría</rdfs:label>

       <rdfs:comment xml:lang="en">A category of project.</rdfs:comment>
       <rdfs:comment xml:lang="fr">Une catégorie de projet.</rdfs:comment>
       <rdfs:comment xml:lang="es">Una categoría de proyecto.</rdfs:comment>

       <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#license">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

       <rdfs:label xml:lang="en">license</rdfs:label>
       <rdfs:label xml:lang="fr">licence</rdfs:label>
       <rdfs:label xml:lang="es">licencia</rdfs:label>

       <rdfs:comment xml:lang="en">The URI of an RDF description of the license
the software is distributed under.</rdfs:comment>
       <rdfs:comment xml:lang="fr">L'URI d'une description RDF de la licence
sous laquelle le programme est distribué.</rdfs:comment>
       <rdfs:comment xml:lang="es">El URI de una descripción RDF de la licencia
bajo la cuál se distribuye el software.</rdfs:comment>
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#repository">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

       <rdfs:label xml:lang="en">repository</rdfs:label>
       <rdfs:label xml:lang="fr">dépôt</rdfs:label>
       <rdfs:label xml:lang="es">repositorio</rdfs:label>

       <rdfs:comment xml:lang="en">Source code repository.</rdfs:comment>
       <rdfs:comment xml:lang="fr">Dépôt du code source.</rdfs:comment>
       <rdfs:comment xml:lang="es">Repositorio del código
fuente.</rdfs:comment>

       <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />

       <rdfs:range rdf:resource="http://usefulinc.com/ns/doap#Repository" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#anon-root">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

       <rdfs:label xml:lang="en">anonymous root</rdfs:label>
       <rdfs:label xml:lang="fr">racine anonyme</rdfs:label>
       <rdfs:label xml:lang="es">raíz anónima</rdfs:label>

       <rdfs:comment xml:lang="en">Repository for anonymous
access.</rdfs:comment>
```

```
	<rdfs:comment xml:lang="fr">Dépôt pour accès anonyme.</rdfs:comment>
	<rdfs:comment xml:lang="es">Repositorio para acceso 
anónimo.</rdfs:comment>

	<rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Repository" />
	<rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal" 
/>
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#browse">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">browse</rdfs:label>
	<rdfs:label xml:lang="fr">visualiser</rdfs:label>
	<rdfs:label xml:lang="es">navegar</rdfs:label>

	<rdfs:comment xml:lang="en">Web browser interface to 
repository.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Interface web au dépôt.</rdfs:comment>
	<rdfs:comment xml:lang="es">Interface web del 
repositorio.</rdfs:comment>

	<rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Repository" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#module">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">module</rdfs:label>
	<rdfs:label xml:lang="fr">module</rdfs:label>
	<rdfs:label xml:lang="es">módulo</rdfs:label>

	<rdfs:comment xml:lang="en">Module name of a CVS, BitKeeper or Arch 
repository.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Nom du module d'un dépôt CVS, BitKeeper ou 
Arch.</rdfs:comment>
	<rdfs:comment xml:lang="es">Nombre del módulo de un repositorio CVS, 
BitKeeper o Arch.</rdfs:comment>

	<!-- doesn't apply to subversion repositories -->
	<rdfs:domain>
	  <owl:Class>
		<owl:unionOf rdf:parseType="Collection">
		  <owl:Class 
rdf:about="http://usefulinc.com/ns/doap#CVSRepository" />
		  <owl:Class 
rdf:about="http://usefulinc.com/ns/doap#ArchRepository" />
		  <owl:Class 
rdf:about="http://usefulinc.com/ns/doap#BKRepository" />
		</owl:unionOf>
	  </owl:Class>
	</rdfs:domain>
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#location">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">repository location</rdfs:label>
	<rdfs:label xml:lang="fr">emplacement du dépôt</rdfs:label>
	<rdfs:label xml:lang="es">lugar del respositorio</rdfs:label>

	<rdfs:comment xml:lang="en">Location of a repository.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Emplacement d'un dépôt.</rdfs:comment>
	<rdfs:comment xml:lang="es">lugar de un repositorio.</rdfs:comment>

	<rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Repository" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#download-page">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">download page</rdfs:label>
	<rdfs:label xml:lang="fr">page de téléchargement</rdfs:label>
```

```
       <rdfs:label xml:lang="es">página de descarga</rdfs:label>

       <rdfs:comment xml:lang="en">Web page from which the project software can
be downloaded.</rdfs:comment>
       <rdfs:comment xml:lang="fr">Page web à partir de laquelle on peut
télécharger le programme.</rdfs:comment>
       <rdfs:comment xml:lang="es">Página web de la cuál se puede bajar el
software.</rdfs:comment>

       <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#download-mirror">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

       <rdfs:label xml:lang="en">download mirror</rdfs:label>
       <rdfs:label xml:lang="fr">miroir pour le téléchargement</rdfs:label>
       <rdfs:label xml:lang="es">mirror de descarga</rdfs:label>

       <rdfs:comment xml:lang="en">Mirror of software download web
page.</rdfs:comment>
       <rdfs:comment xml:lang="fr">Miroir de la page de téléchargement du
programme.</rdfs:comment>
       <rdfs:comment xml:lang="es">Mirror de la página web de
descarga.</rdfs:comment>

       <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#revision">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

       <rdfs:label xml:lang="en">revision</rdfs:label>
       <rdfs:label xml:lang="fr">révision</rdfs:label>
       <rdfs:label xml:lang="es">versión</rdfs:label>

       <rdfs:comment xml:lang="en">Revision identifier of a software
release.</rdfs:comment>
       <rdfs:comment xml:lang="fr">Identifiant de révision d'une release du
programme.</rdfs:comment>
       <rdfs:comment xml:lang="es">Indentificador de la versión de un release
de software.</rdfs:comment>

       <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Version" />

       <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"
/>
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#file-release">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

       <rdfs:label xml:lang="en">file-release</rdfs:label>

       <rdfs:comment xml:lang="en">URI of download associated with this
release.</rdfs:comment>

       <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Version" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#wiki">
       <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

       <rdfs:label xml:lang="en">wiki</rdfs:label>
       <rdfs:label xml:lang="fr">wiki</rdfs:label>
       <rdfs:label xml:lang="es">wiki</rdfs:label>

       <rdfs:comment xml:lang="en">URL of Wiki for collaborative discussion of
project.</rdfs:comment>
       <rdfs:comment xml:lang="fr">L'URL du Wiki pour la discussion
collaborative sur le projet.</rdfs:comment>
       <rdfs:comment xml:lang="es">URL del Wiki para discusión colaborativa del
proyecto.</rdfs:comment>
```

```
        <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#bug-database">
        <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

        <rdfs:label xml:lang="en">bug database</rdfs:label>
        <rdfs:label xml:lang="fr">suivi des bugs</rdfs:label>
        <rdfs:label xml:lang="es">base de datos de bugs</rdfs:label>

        <rdfs:comment xml:lang="en">Bug tracker for a project.</rdfs:comment>
        <rdfs:comment xml:lang="fr">Suivi des bugs pour un
projet.</rdfs:comment>
        <rdfs:comment xml:lang="es">Bug tracker para un proyecto.</rdfs:comment>

        <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#screenshots">
        <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

        <rdfs:label xml:lang="en">screenshots</rdfs:label>
        <rdfs:label xml:lang="fr">captures d'écran</rdfs:label>
        <rdfs:label xml:lang="es">capturas de pantalla</rdfs:label>

        <rdfs:comment xml:lang="en">Web page with screenshots of
project.</rdfs:comment>
        <rdfs:comment xml:lang="fr">Page web avec des captures d'écran du
projet.</rdfs:comment>
        <rdfs:comment xml:lang="es">Página web con capturas de pantalla del
proyecto.</rdfs:comment>

        <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#maintainer">
        <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

        <rdfs:label xml:lang="en">maintainer</rdfs:label>
        <rdfs:label xml:lang="fr">développeur principal</rdfs:label>
        <rdfs:label xml:lang="es">desarrollador principal</rdfs:label>

        <rdfs:comment xml:lang="en">Maintainer of a project, a project
leader.</rdfs:comment>
        <rdfs:comment xml:lang="fr">Développeur principal d'un projet, un meneur
du projet.</rdfs:comment>
        <rdfs:comment xml:lang="es">Desarrollador principal de un proyecto, un
líder de proyecto.</rdfs:comment>

        <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />

        <rdfs:range rdf:resource="http://xmlns.com/foaf/0.1/Person" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#developer">
        <rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

        <rdfs:label xml:lang="en">developer</rdfs:label>
        <rdfs:label xml:lang="fr">développeur</rdfs:label>
        <rdfs:label xml:lang="es">desarrollador</rdfs:label>

        <rdfs:comment xml:lang="en">Developer of software for the
project.</rdfs:comment>
        <rdfs:comment xml:lang="fr">Développeur pour le projet.</rdfs:comment>
        <rdfs:comment xml:lang="es">Desarrollador de software para el
proyecto.</rdfs:comment>

        <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />

        <rdfs:range rdf:resource="http://xmlns.com/foaf/0.1/Person" />
</rdf:Property>
```

```
<rdf:Property rdf:about="http://usefulinc.com/ns/doap#documenter">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">documenter</rdfs:label>
	<rdfs:label xml:lang="fr">rédacteur de l'aide</rdfs:label>
	<rdfs:label xml:lang="es">escritor de ayuda</rdfs:label>

	<rdfs:comment xml:lang="en">Contributor of documentation to the
project.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Collaborateur à la documentation du
projet.</rdfs:comment>
	<rdfs:comment xml:lang="es">Proveedor de documentación para el
proyecto.</rdfs:comment>

	<rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />

	<rdfs:range rdf:resource="http://xmlns.com/foaf/0.1/Person" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#translator">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">translator</rdfs:label>
	<rdfs:label xml:lang="fr">traducteur</rdfs:label>
	<rdfs:label xml:lang="es">traductor</rdfs:label>

	<rdfs:comment xml:lang="en">Contributor of translations to the
project.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Collaborateur à la traduction du
projet.</rdfs:comment>
	<rdfs:comment xml:lang="es">Proveedor de traducciones al
proyecto.</rdfs:comment>

	<rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />

	<rdfs:range rdf:resource="http://xmlns.com/foaf/0.1/Person" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#tester">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">tester</rdfs:label>
	<rdfs:label xml:lang="fr">testeur</rdfs:label>
	<rdfs:label xml:lang="es">tester</rdfs:label>

	<rdfs:comment xml:lang="en">A tester or other quality control
contributor.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Un testeur ou un collaborateur au contrôle
qualité.</rdfs:comment>
	<rdfs:comment xml:lang="es">Un tester u otro proveedor de control de
calidad.</rdfs:comment>

	<rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />

	<rdfs:range rdf:resource="http://xmlns.com/foaf/0.1/Person" />
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#helper">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">helper</rdfs:label>
	<rdfs:label xml:lang="fr">collaborateur</rdfs:label>
	<rdfs:label xml:lang="es">colaborador</rdfs:label>

	<rdfs:comment xml:lang="en">Project contributor.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Collaborateur au projet.</rdfs:comment>
	<rdfs:comment xml:lang="es">Colaborador del proyecto.</rdfs:comment>

	<rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />

	<rdfs:range rdf:resource="http://xmlns.com/foaf/0.1/Person" />
</rdf:Property>
```

```
<rdf:Property rdf:about="http://usefulinc.com/ns/doap#programming-language">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">programming language</rdfs:label>
	<rdfs:label xml:lang="fr">langage de programmation</rdfs:label>
	<rdfs:label xml:lang="es">lenguaje de programación</rdfs:label>

	<rdfs:comment xml:lang="en">Programming language a project is
implemented in or intended for use with.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Langage de programmation avec lequel un
projet est implémenté, ou avec lequel il est prévu de
l'utiliser.</rdfs:comment>
	<rdfs:comment xml:lang="es">Lenguaje de programación en el que un
proyecto es implementado o con el cuál pretende usarse.</rdfs:comment>

	<rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />

	<rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"
/>
</rdf:Property>

<rdf:Property rdf:about="http://usefulinc.com/ns/doap#os">
	<rdfs:isDefinedBy rdf:resource="http://usefulinc.com/ns/doap#" />

	<rdfs:label xml:lang="en">operating system</rdfs:label>
	<rdfs:label xml:lang="fr">système d'exploitation</rdfs:label>
	<rdfs:label xml:lang="es">sistema operativo</rdfs:label>

	<rdfs:comment xml:lang="en">Operating system that a project is limited
to.  Omit this property if the project is not OS-specific.</rdfs:comment>
	<rdfs:comment xml:lang="fr">Système d'exploitation auquel est limité le
projet. Omettez cette propriété si le projet n'est pas limité à un système
d'exploitation.</rdfs:comment>
	<rdfs:comment xml:lang="es">Sistema opertivo al cuál está limitado el
proyecto.  Omita esta propiedad si el proyecto no es específico de un sistema
opertaivo en particular.</rdfs:comment>

	<rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project" />

	<rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"
/>
</rdf:Property>

</rdf:RDF>
```

Appendix 1 - The SWDOAP Vocabulary

```
<?xml version="1.0" encoding='ISO-8859-3'?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xml:base="file:swdoap.rdf"
  >

  <owl:Ontology rdf:about="file:swdoap.rdf">
    <owl:imports rdf:resource="http://usefulinc.com/ns/doap"/>
    <owl:imports rdf:resource="http://www.w3.org/2000/10/annotationType"/>
    <owl:imports rdf:resource="http://www.w3.org/2000/10/annotation-ns"/>

    <dc:title>The extended DOAP vocabulary for Semantic Web
Applications</dc:title>
    <dc:description>The extended Description of a Project (DOAP) vocabulary
for Semantic Web Applications, described using W3C RDF Schema and the Web
Ontology Language.</dc:description>
    <dc:creator>Leonardo Magela Cunha</dc:creator>
    <dc:format>application/rdf+xml</dc:format>

    <foaf:maker>
              <foaf:Person>
                     <foaf:name>Leonardo Magela Cunha</foaf:name>
                     <foaf:mbox
rdf:resource="mailto:leomagela+swdoap@gmail.com" />
              </foaf:Person>
    </foaf:maker>

  </owl:Ontology>

  <!-- Classes -->

  <rdfs:Class rdf:ID="Ontology">
       <rdfs:isDefinedBy rdf:resource="" />
    <rdfs:comment xml:lang="en">An ontology.</rdfs:comment>
    <rdfs:label xml:lang="en">Ontology</rdfs:label>
  </rdfs:Class>

  <rdfs:Class rdf:about="QueryDescriptionLanguage">
       <rdfs:isDefinedBy rdf:resource="" />
        <rdfs:comment xml:lang="en">An ontology query description
language.</rdfs:comment>
    <rdfs:label xml:lang="en">Query Description Language</rdfs:label>
  </rdfs:Class>

  <rdfs:Class rdf:about="Category">
       <rdfs:isDefinedBy rdf:resource="" />
    <rdfs:comment xml:lang="en">A category from a classification and
categorization system, such as Trove.</rdfs:comment>
    <rdfs:label xml:lang="en">Category</rdfs:label>
  </rdfs:Class>

  <rdfs:Class rdf:about="DistributionMethod">
       <rdfs:isDefinedBy rdf:resource="" />
    <rdfs:comment xml:lang="en">A distribuition method.</rdfs:comment>
    <rdfs:label xml:lang="en">Distribution Method</rdfs:label>
  </rdfs:Class>

  <rdfs:Class rdf:about="SupportingTech">
       <rdfs:isDefinedBy rdf:resource="" />
    <rdfs:label xml:lang="en">Supporting Technology</rdfs:label>
    <rdfs:comment xml:lang="en">A supporting technology.</rdfs:comment>
  </rdfs:Class>

  <rdfs:Class rdf:about="SoftwareComponentType">
       <rdfs:isDefinedBy rdf:resource="" />
    <rdfs:comment xml:lang="en">The type of a software component. For example:
agent, component etc.</rdfs:comment>
    <rdfs:label xml:lang="en">Software Component Type</rdfs:label>
```

```
  </rdfs:Class>

  <rdfs:Class rdf:about="PersistenceTech">
      <rdfs:isDefinedBy rdf:resource="" />
    <rdfs:label xml:lang="en">Persistence Technology</rdfs:label>
    <rdfs:comment xml:lang="en">A persistence technology.</rdfs:comment>
  </rdfs:Class>

  <rdfs:Class rdf:about="DescriptionLanguage">
      <rdfs:isDefinedBy rdf:resource="" />
    <rdfs:comment xml:lang="en">An ontology description
language.</rdfs:comment>
    <rdfs:label xml:lang="en">Description Language</rdfs:label>
  </rdfs:Class>

  <rdfs:Class rdf:about="ReasoningTech">
      <rdfs:isDefinedBy rdf:resource="" />
    <rdfs:label xml:lang="en">Reasoning Technology</rdfs:label>
    <rdfs:comment xml:lang="en">A reasoning technology.</rdfs:comment>
  </rdfs:Class>

  <!-- Properties -->
  <rdf:Property rdf:about="same-purpose-as-original">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:label xml:lang="en">Same Purpose As Orginal</rdfs:label>
    <rdfs:comment xml:lang="en">Is the data, manipulated by the project, used
in a different purpose than original?</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="information-sources-observation">
    <rdfs:comment xml:lang="en">Observation about the information sources used
by the project.</rdfs:comment>
    <rdfs:label xml:lang="en">Information Sources Observation</rdfs:label>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
  </rdf:Property>

  <rdf:Property rdf:about="afilliation">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:label xml:lang="en">Afilliation</rdfs:label>
    <rdfs:comment xml:lang="en">The afilliation of a Project.</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="ontology">
    <rdfs:label xml:lang="en">Ontology</rdfs:label>
    <rdfs:comment xml:lang="en">An ontology used by a project.</rdfs:comment>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:range rdf:resource="#Ontology"/>
  </rdf:Property>

  <rdf:Property rdf:about="structurally-heterogenous">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:label xml:lang="en">Structurally Heterogenous</rdfs:label>
    <rdfs:comment xml:lang="en">Does the project organize information in
different ways?</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="real-world-data">
    <rdfs:comment xml:lang="en">Does the project use real world
data?</rdfs:comment>
    <rdfs:label xml:lang="en">Real World Data Usage</rdfs:label>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
  </rdf:Property>

  <rdf:Property rdf:about="audience-type">
    <rdfs:comment xml:lang="en">Whom are the final users?</rdfs:comment>
    <rdfs:label xml:lang="en">Audience Type</rdfs:label>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
  </rdf:Property>

  <rdf:Property rdf:about="syntatically-heterogenous">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:label xml:lang="en">Syntatically Heterogenous</rdfs:label>
```

```
    <rdfs:comment xml:lang="en">Does the proejct use different syntatic
standards?</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="diverse-ownership">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:label xml:lang="en">Information Sources' Diverse
Ownership</rdfs:label>
    <rdfs:comment xml:lang="en">Does the information sources of this project
have diverse ownership?</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="scalable-in-number-of-components">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:label xml:lang="en">Scalable in Number of Components</rdfs:label>
    <rdfs:comment xml:lang="en">Is the project scalable in the number of
components used?</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="persistenceTech">
    <rdfs:label xml:lang="en">Persistence Technology</rdfs:label>
    <rdfs:range rdf:resource="#PersistenceTech"/>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:comment xml:lang="en">A persistence technology used by a
project.</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="multiple-language">
    <rdfs:label xml:lang="en">Multiple Language Supported</rdfs:label>
    <rdfs:comment xml:lang="en">Does the information sources of this project
support multiple languages?</rdfs:comment>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
  </rdf:Property>

  <rdf:Property rdf:about="semantically-heterogenous">
    <rdfs:label xml:lang="en">Semantically Heterogenous</rdfs:label>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >does the project use different terminologies to refer to the same
information?</rdfs:comment>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
  </rdf:Property>

  <rdf:Property rdf:about="metadata-observation">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:comment xml:lang="en">Observation about the metadata about this
project.</rdfs:comment>
    <rdfs:label xml:lang="en">Metadata Observation</rdfs:label>
  </rdf:Property>

  <rdf:Property rdf:about="data-domain">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:label xml:lang="en">Data Domain</rdfs:label>
    <rdfs:comment xml:lang="en">What is the domain of data?</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="softwareComponentType">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:comment xml:lang="en">A software component type of a
project</rdfs:comment>
    <rdfs:range rdf:resource="#SoftwareComponentType"/>
    <rdfs:label xml:lang="en">Software Component Type</rdfs:label>
  </rdf:Property>

  <rdf:Property rdf:about="last-visited">
    <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:label xml:lang="en">Last Visit</rdfs:label>
    <rdfs:comment xml:lang="en">Date of the last visit to the homepage of a
project, in YYYY-MM-DD form. e.g. 2004-04-05.</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="descriptionLanguage">
    <rdfs:label xml:lang="en">Description Language</rdfs:label>
```

```
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:comment xml:lang="en">An ontology description language used by a
project.</rdfs:comment>
    <rdfs:range rdf:resource="#DescriptionLanguage"/>
  </rdf:Property>

  <rdf:Property rdf:about="distributionMethod">
    <rdfs:range rdf:resource="#DistributionMethod"/>
    <rdfs:comment xml:lang="en">A distribution method used by a
project.</rdfs:comment>
    <rdfs:label xml:lang="en">Distribution Method</rdfs:label>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
  </rdf:Property>

  <rdf:Property rdf:about="distributed">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:label xml:lang="en">Distributed</rdfs:label>
    <rdfs:comment xml:lang="en">Are the information sources of the project
distributed?</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="doap-url">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:label xml:lang="en">DOAP URL</rdfs:label>
    <rdfs:comment xml:lang="en">The DOAP URL of a project.</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="multimedia">
    <rdfs:comment xml:lang="en">Does the project us the content of multimedia
documents? Semantically?</rdfs:comment>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:label xml:lang="en">Multimedia Documents Usage</rdfs:label>
  </rdf:Property>

  <rdf:Property rdf:about="challenge-ranking">
    <rdfs:label xml:lang="en">SWC Ranking</rdfs:label>
    <rdfs:comment xml:lang="en">Ranking reached by a project in the Semantic
Web Challenge</rdfs:comment>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
  </rdf:Property>

  <rdf:Property rdf:about="contact">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:range rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
    <rdfs:label xml:lang="en">Contact</rdfs:label>
    <rdfs:comment xml:lang="en">A contact of a project</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="data-meaning-observation">
    <rdfs:label xml:lang="en">Data Meaning Observation</rdfs:label>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:comment xml:lang="en">Observation abou the use of data meaning done
by the project.</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="challenge-year">
    <rdfs:label xml:lang="en">SWC's Submission Year</rdfs:label>
    <rdfs:comment xml:lang="en">Year of the submission of the projecto to the
Semantic Web Challenge.</rdfs:comment>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
  </rdf:Property>

  <rdf:Property rdf:about="open-source">
    <rdfs:label xml:lang="en">Open Source</rdfs:label>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:comment xml:lang="en">By the DOAP definition, it was supposed to be
a schema to describe open source projects. However, this is not the case for
the projects of SWC. This property is the intended to explicit if a project is
open source or not.</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="scalable">
    <rdfs:label xml:lang="en">Scalable</rdfs:label>
```

```
    <rdfs:comment xml:lang="en">How many data sources are used?</rdfs:comment>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
  </rdf:Property>

  <rdf:Property rdf:about="queryDescriptionLanguage">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:comment xml:lang="en">An ontology query description language used by
a project.</rdfs:comment>
    <rdfs:label xml:lang="en">Query Description Language</rdfs:label>
    <rdfs:range rdf:resource="#QueryDescriptionLanguage"/>
  </rdf:Property>

  <rdf:Property rdf:about="application-observation">
    <rdfs:comment xml:lang="en">Observation abou the applications aspect of
the project.</rdfs:comment>
    <rdfs:label xml:lang="en">Application Observation</rdfs:label>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
  </rdf:Property>

  <rdf:Property rdf:about="diverse-method-of-access">
    <rdfs:label xml:lang="en">Diverse Method of Access</rdfs:label>
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:comment xml:lang="en">Does the project support diverse methods of
access? For example, mobile access.</rdfs:comment>
  </rdf:Property>

  <rdf:Property rdf:about="supporting-tech">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:comment xml:lang="en">Supporting technology used by the
project.</rdfs:comment>
    <rdfs:label xml:lang="en">Supporting Technology</rdfs:label>
    <rdfs:range rdf:resource="#SupportingTech"/>
  </rdf:Property>

  <rdf:Property rdf:about="reasoningTech">
    <rdfs:domain rdf:resource="http://usefulinc.com/ns/doap#Project"/>
    <rdfs:comment xml:lang="en">A reasoning technology used by a
project.</rdfs:comment>
    <rdfs:label xml:lang="en">Reasoning Technology</rdfs:label>
    <rdfs:range rdf:resource="#ReasoningTech"/>
  </rdf:Property>
</rdf:RDF>
```