Gerenciamento de Anotações Semânticas de Dados na Web para Aplicações Agrícolas

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Sidney Roberto de Sousa e aprovada pela Banca Examinadora.

Campinas, 12 de março de 2010.

Prof.ª Dr.ª Claudia Bauzer Medeiros
Instituto de Computação - UNICAMP
(Orientadora)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

FICHA CATALOGRÁFICA ELABORADA
PELA
BIBLIOTECA DO IMECC DA UNICAMP
Bibliotecária: Maria Fabiana Bezerra Müller – CRB8 / 6162

Sousa, Sidney Roberto de

So85g Gerenciamento de anotações semânticas de dados na Web para aplicações agrícolas/Sidney Roberto de Sousa-- Campinas, [S.P. : s.n.], 2010.

Orientador : Claudia Bauzer Medeiros

Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Computação.

1.Semântica - Processamento de dados. 2.Ontologia. 3.Recuperação da informação. 4.Web semântica. I. Medeiros, Claudia Maria Bauzer. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Título em inglês: Management of semantic annotations of data on the Web for agricultural applications

Palavras-chave em inglês (Keywords): 1. Semantics – Data processing. 2. Ontology. 3. Information retrieval. 4.Semantic Web.

Área de concentração: Banco de Dados

Titulação: Mestre em Ciência da Computação

Banca examinadora: Profª Drª Claudia Bauzer Medeiros (IC-UNICAMP)
Prof. Dr. Andre Santanchè (UNIFACS)
Prof. Dr. Ronaldo dos Santos Mello (INE-UFSC)

Data da defesa: 12/03/2010

Programa de Pós-Graduação: Mestrado em Ciência da Computação

TERMO DE APROVAÇÃO

Dissertação Defendida e Aprovada em 12 de março de 2010, pela Banca examinadora composta pelos Professores Doutores:

Ronaldo dos Santos Mello

Prof. Dr. Ronaldo dos Santos Mello
Departamento de Informática e Estatística / UFSC

André Santanchè

Prof. Dr. André Santanchè
Departamento de Ciências Exatas / UNIFACS

Claudia Bauzer Medeiros

Prof^{a}. Dr^{a}. Claudia Maria Bauzer Medeiros
IC / UNICAMP

Instituto de Computação
Universidade Estadual de Campinas

Gerenciamento de Anotações Semânticas de Dados na Web para Aplicações Agrícolas

Sidney Roberto de Sousa[1]

Março de 2010

Banca Examinadora:

- Prof.ª Dr.ª Claudia Bauzer Medeiros
  Instituto de Computação - UNICAMP (Orientadora)
- Prof. Dr. Andre Santanchè
  UNIFACS
- Prof. Dr. Ronaldo dos Santos Mello
  Departamento de Informática e Estatística - UFSC
- Prof. Dr. Sandro Rigo (Suplente)
  Instituto de Computação - UNICAMP
- Prof. Dr. Rubens Lamparelli (Suplente)
  CEPAGRI - UNICAMP

[1] Suporte financeiro de: FAPESP (processo 2008/51028-4), além do Instituto Virtual FAPESP-Microsoft Research.

Resumo

Sistemas de informação geográfica a cada vez mais utilizam informação geo-espacial da *Web* para produzir informação geográfica. Um grande desafio para tais sistemas é encontrar dados relevantes, onde tal busca é frequentemente baseada em palavras-chave ou nome de arquivos. Porém, tais abordagens carecem de semântica. Desta forma, torna-se necessário oferecer mecanismos para preparação de dados, afim de auxiliar a recuperação de dados semanticamente relevantes. Para atacar este problema, esta dissertação de mestrado propôem uma arquitetura baseada em serviços para gerenciar anotações semânticas. Neste trabalho, uma anotação semântica é um conjunto de triplas - chamadas unidades de anotação semântica - $< subject, metadata\ field, object >$, onde $subject$ é um documento geo-espacial, $($metadata field) é um campo de metadados sobre este documento e *object* é um termo de ontologia que associa semanticamente o campo de metadados a algum conceito apropriado.

As principais contribuições desta dissertação são: um estudo comparativo sobre ferramentas de anotação; especificação e implementação de uma arquitetura baseada em serviços para gerenciar anotações semânticas, incluindo serviços para manuseio de termos de ontologias; e uma análise comparativa de mecanismos para armazenar anotações semânticas. O trabalho toma como estudo de caso anotações semânticas sobre documentos agrícolas.

Abstract

Geographic information systems (GIS) are increasingly using geospatial data from the Web to produce geographic information. One big challenge is to find the relevant data, which often is based on keywords or even file names. However, these approaches lack semantics. Thus, it is necessary to provide mechanisms to prepare data to help retrieval of semantically relevant data. To attack this problem, this dissertation proposes a service-based architecture to manage semantic annotations. In this work, a semantic annotation is a set of triples - called semantic annotation units - $< subject, metadata\ field, object >$, where *subject* is a geospatial resource, $($metadata field) contains some characteristic about this resource, and *object* is an ontology term that semantically associates the metadata field to some appropriate concept.

The main contributions of this dissertation are: a comparative study on annotation tools; specification and implementation of a service-based architecture to manage semantic annotations, including services for handling ontology terms; and a comparative analysis of mechanisms for storing semantic annotations. The work takes as case study semantic annotations about agricultural resources.

Agradecimentos

À Claudia, pela orientação com pulso firme, que me fez quebrar muitas das barreiras da minha capacidade intelectual. Além disso, agradeço por se preocupar não somente com a minha dissertação de mestrado, mas também com o meu futuro profissional.

À minha esposa Suelen, pelo companheirismo sem limites e o amor incondicional. Sem você eu não teria chegado a lugar algum, pois você mais do que ninguém me faz sentir uma pessoal especial!

Aos meus pais Antônio e Terezinha e meus irmãos Fábio e Aninha, por sempre me julgarem como uma pessoa sem limites intelectuais que pode voar sempre mais e mais alto. Vocês me fazem sentir mais vivo a cada dia!

Aos amigos do LIS, que sempre estiveram dispostos a tirar as minhas dúvidas e me auxiliar com os meus problemas, sem se importar com o tamanho e a complexidade destes.

Aos membros da banca, professores Ronaldo Mello e André Santanchè, pelas sugestões.

À FAPESP (processo 08/51028-4) e ao Microsoft Research - FAPESP Institute for ICT Research (projeto eFarms), pelo apoio financeiro.

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

A *Web* se tornou um imenso repositório de dados geo-espaciais em diferentes formatos geográficos como imagens de sensoriamento remoto, mapas, séries temporais de dados de sensores, arquivos de texto, entre outros [23, 48]. A recuperação de tais dados requer uma atenção especial devido à sua heterogeneidade e a distribuição geográfica de suas fontes. Padrões de metadados geográficos e portais de informações geo-espaciais foram criados como uma iniciativa para atacar este problema.

Em tais portais, os usuários podem criar suas próprias consultas utilizando palavras-chave e campos de metadados, usando algum esquema de metadados como por exemplo os propostos pela ISO 19115 [32] ou o padrão de metadados da FGDC [20]. Nesta abordagem, os campos de metadados são preenchidos com texto em linguagem natural, o que pode levar a ambiguidades. Além disto, o uso de palavras-chave pode restringir o resultado das consultas caso sejam usadas diferentes terminologias ou se os termos forem homônimos [38].

Uma solução para resolver tais problemas é o uso de ontologias de domínio - como pode ser visto em [37] - para identificar e associar conceitos. Ontologias são frequentemente utilizadas para explicar conhecimento sobre algum domínio de interesse. Dentro do domínio geográfico, uma ontologia precisa possuir termos e conceitos úteis para descrever documentos digitais geo-espaciais, como por exemplo referências geo-espaciais, períodos temporais, detalhes sobre formatos geográficos além de outros tipos de meta-informação que visam otimizar a recuperação de informação geo-espacial.

O *World Wide Web Consortium* (W3C) propôs o *Resource Description Framework* (RDF) [84] para descrever documentos disponíveis na *Web* como uma iniciativa para oferecer interoperabilidade semântica. RDF identifica documentos utilizando suas URL's e os descreve utilizando declarações (*statements*). Uma declaração é uma tripla <*subject, predicate, object*>. Pastorello Jr. [60] adaptou esta definição para definir o conceito de unidade de anotação, em que *subject* é um objeto digital, *predicate* é um campo de metadados deste documento e *object* é o valor preenchido neste campo de metadados. Desta forma, para efeito da dissetação, o conjunto de unidades de anotação com *subject* em comum é uma anotação.

Aplicando este modelo de tal forma que ontologias possam ser inclusas, *object* pode ser um termo de ontologia o qual associa semanticamente o campo de metadados a algum conceito apropriado e temos em consequência uma *unidade de anotação semântica*. Assim, o conjunto de unidades de anotação semântica que possuem *subject* em comum compôem uma *anotação semântica*.

Baseada nesta abordagem, esta dissertação propôe uma arquitetura para a transformação de anotações em anotações semânticas, em que *subject* seja restrito a documentos com conteúdo geo-espacial. A arquitetura oferece serviços para a criação, armazenamento e recuperação de anotações semânticas. O processo de criação destas anotações semânticas acarretou o desenvolvimento de um repositório de termos de ontologias, além de um serviço para popular tal repositório. A dissertação toma como estudo de caso documentos com informações úteis para a tomada de decisão na agricultura, tendo em vista a forte presença de conteúdo geo-referenciado nestes e a utilidade de sua aplicação.

As principais contribuições desta dissertação são:

- Estudo comparativo sobre ferramentas de anotação de dados, considerando diferentes naturezas de anotação;

- Especificação e implementação de algoritmos para processar e ordenar por relevância termos de ontologias, afim de permitir diferentes modalidades de gerenciamento de anotações semânticas;

- Especificação e implementação de uma arquitetura baseada em serviços para geração e gestão de anotações semânticas, incluindo vários serviços de manuseio de termos de ontologias;

- Análise comparativa de mecanismos para armazenamento de anotações semânticas.

Parte da pesquisa foi publicada nos seguintes artigos:

- S. R. Sousa and C. B. Medeiros. Management of Semantic Annotations of Data on the Web for Agricultural Applications. VIII Workshop de Teses e Dissertações em Bancos de Dados, 2009 [16];

- S. R. Sousa. A Semantic Approach to Describe Geospatial Resources. LNCS - 3rd International Workshop on Semantic and Conceptual Issues in GIS (SeCoGIS 2009), vol. 5833, pg. 327-336, 2009 [15];

- C. G. N. Macário, S. R. de Sousa, and C. B. Medeiros. Annotating Geospatial Data based on its Semantics. 17th ACM SIGSPATIAL Conference, p.81-90, 2009 [47].

O texto desta dissertação está organizado da seguinte forma. O Capítulo 2 apresenta conceitos abordados na pesquisa e trabalhos relacionados. O Capítulo 3 apresenta a arquitetura proposta para o gerenciamento de anotações semânticas, discutindo as soluções e algoritmos utilizados. O Capítulo 4 apresenta os aspectos de implementação da arquitetura, além de um exemplo de seu uso, testes e validações. Por fim, o Capítulo 5 contém conclusões e trabalhos futuros.

Chapter 2

Related Work and Basic Concepts

This chapter presents the basic concepts used in this dissertation, and related work. Section 2.1 describes semantic annotations and their use in information systems. Section 2.2 presents some common metadata standards. Section 2.3 describes ontologies and technologies involved in the creation thereof. Section 2.4 presents the eFarms project and its association to this dissertation. Section 2.5 discusses research that applies semantics on agricultural information. Section 2.6 presents a survey on different works on annotation tools and approaches, analysing them. Finally, section 2.7 presents conclusions.

2.1 Semantic Annotations

The semantic annotation concept comes from the annotation concept. To annotate means to attach data to some other piece of data [59] - similar to metadata. Annotations describe a resource (digital or not), considering its characteristics. An annotation can be created manually [41], semi-automatically [26], or automatically [9]. Furthermore, a resource can be annotated in various manners, for instance, using free-text, sketches, voice, or drawings, among others - see section 2.6.1.

An annotation can be structured using metadata fields, as shown in [43, 52]. Such fields are often used to organize an annotation by its characteristics. For instance, a digital music file can contain metadata fields about title, artist, album, among other, whereas a remote sensing image can contain fields about geographic region, latitude, longitude, and the satellite that took the image. Like annotations, semantic annotations also can use metadata fields for improving their semantics, as can be seen in [60, 47]. Furthermore, ontologies are often used in semantic annotations to provide a controled vocabulary, as can be seen in [73, 22].

The Semantic Web is an extension of the World Wide Web proposed by Berners-Lee [5], whose goal is to support access to documents directly so that machines can be able to analyse data from the Web and infer useful information. For the full implementation of the Semantic Web, it is necessary that its content be properly classified. A semantic annotation is a description of some digital content document according to its semantics. To semantically annotate a document means to assign to it some information that will allow interpretation of its content. A similar concept is semantic tagging [17], which consists in applying tags that semantically describe a document.

Popov *et al* [61] consider that a semantic annotation corresponds to assigning to the entities in a text links to their semantic descriptions, providing both class and instance information about the entities referenced in the documents. Pastorello Jr. *et al* [60] define a semantic annotation as a set of annotation units $< s, p, o >$, where s is the subject being described, p is a property from this subject, and o represents a describing object or value.

2.2 Metadata Standards

Metadata fields provide organization of descriptions about documents. Absence of metadata may lead to unreliability and re-work when it comes to interoperability among distinct systems, hampering data exchange and integration [55]. Moreover, using metadata fields from little used standards may hamper sharing of documents among different users and applications, since there is no consensus among them. To attack this problem, metadata standards were proposed in order to improve data sharing and integration.

One of the most used standards is Dublin Core [77]. It is a generic standard that aims to provide general description about any document so that it can be easily found and retrieved. The main implementations of Dublin Core are based on RDF [84] or XML. The Dublin Core standard is divided in two levels: Simple Dublin Core, that contains fifteen metadata elements, which are the basic elements needed to describe a document; and Qualified Dublin Core, that contains extra elements to provide more semantics.

Geographic metadata describe geospatial resources, enhancing them with useful information such as the reference system used, producer identification, and location information. The use of geographic metadata is strongly disseminated by geographic catalogs, such as GeoNetwork [24].

ISO 19115 [32] is a geographic metadata standard, developed by the ISO Committee. It has a UML based structure, where each metadata element is defined in context of a class and is characterized by a *name*, *definition*, *obligation*, *multiplicity*, *data type*, and a *domain*. This standard has a minimal set of elements which is defined for the most important information needed to describe some resource, called *core data*. It is possible to extend this set of elements to serve special needs [36].

The Federal Geographic Data Committee Metadata (FGDC Metadata) [20] is an open standard which defines particularities needed to catalog and publish geographic meta-information. It provides knowledge about the kind of the resource, indicating whether it meets the user's expectation, and where/how to find it. Use of a specific section or element is either mandatory or optional [12].

2.3 Ontologies

An ontology is commonly defined as a formal and explicit specification of a shared conceptualization from a domain of interest [29]. It describes a relevant part of the world, in a language that can be understood by machines. An ontology necessarily incorporates some view from the world that refers to a given domain. This view is generally conceived as a set of concepts (e.g.; entities, attributes, processes), their definitions and inter-relationships; this is referred to as a conceptualization [72]. Ontologies are basically composed by:

- **classes**: define concepts from the domain
- **properties**: define characteristics of the classes
- **instances**: individuals or objects of the classes (ground-level objects)
- **relations**: ways in which classes and individuals can be related to each other

According to Gruber [30], ontologies are used to describe ontological "commitments" for a set of agents so that they can talk about a domain of discourse without the need of operating over a globally shared theory. An agent is commited to an ontology if its actions are consistent with the definitions of the ontology.

An ontology language is a formal language used to represent ontologies. Most languages are based on RDF [84], a family of specifications of W3C that was formerly developed as a data model for metadata and has become a general method for modelling information, using a variety of syntactic formats. The language most used to represent ontologies is OWL (recursive acronym for Web Ontology Language), which is based on RDF, RDF Schema, and projects of ontology languages such as OIL [22] and DAML [31]. All OWL elements (instances, classes, and properties) are defined as RDF resources and identified by URIs (Uniform Resurce Identification).

Data described by an OWL ontology is interpreted as a set of classes and a set of properties that connect those classes to each other. Ontology axioms allow the discovery of additional facts. For instance, an ontology describing agricultural crops may include axioms indicating that a *hasSameHarvestPeriod* property can only exist between two individuals when a *hasSamePlantationPeriod* property also exists between them.

2.4 The eFarms Project

eFarms [39] is a multidisciplinary project financed by the FAPESP-Microsoft Research Virtual Institute that combines research on computer science and agricultural sciences. Its aim is to deal with practical and theoretical problems involving management of agricultural data and low-cost wireless communication in rural areas in Brazil.

Based on these problems, the project has three main objectives:

- **Wireless data communication**: to perform management of data, integrating different wireless technologies. Using these technologies, the project aims to make possible a real-time communication between farmers, which will be able to be not just farmers, but active data providers;
- **Data fusion and analysis**: integration of heterogeneous data and support to semantic context. The project aims to handle integration of remote sensing data, image processing, and methodologies for software developing. At this context, it aims to investigate research issues like data fusion and semantic annotation;
- **Proposal of models and implementation**: to develop and build new models of crop productivity forecasting from the point of view of agricultural sciences, thus giving support to decision making in crops management. In this level, the project concerns about specification of models, implementation, and tests.

The work presented in this dissertation has been developed in association with a PhD thesis [46]. It contributes to the data analysis and fusion level, providing a tool for semantic annotation of data resources used in agricultural planning, especially distinct kinds of satellite image products and time series from sensors. The work of [46] provides workflows for acquisition of meta-information of agricultural documents, creating annotations about them. The work of this dissertation transforms such annotations into semantic annotations, storing them for future use.

2.5 Semantics in Agriculture

There have been several initiatives concerning the development of applications in agriculture that take semantics into account. FAO (Food and Agriculture Organization) - jointly with the Commission of the European Communities - has developed AGROVOC [42], a multilingual thesaurus of agricultural and ecological terms. It is widely used for indexing and retrieving data from agricultural information systems. It has free license for non-commercial uses and is available in several formats like RDF, OWL, MySQL, and MSAccess. The AGROVOC structure is composed by terms which consists of one or more words representing a concept. For each term, a set of words is shown, describing its relationship to other terms. Furthermore, scope notes are used for making clear the meaning and the context of the terms. Taxonomical and geographical terms are tagged to facilitate their retrieval.

There are several papers that focus on ontologies for the agricultural domain. The work of [50] describes an ontology for horticulture and vegetable crops. It was built using the Protégé ontology editor [65] and contains terms about crop techniques, management of plagues, harvesting, and vegetables consumption and selling. Its aim is to provide knowledge extraction about fish species cultivated in low-technology greenhouses located on the Mediterranean basis and Greece. This ontology was developed to be used by different application areas. Thus, it provides translation of the notation that it uses to other notations that specific applications may need.

The work of [3] proposes a method for developing ontologies using semantic information available in the titles of digital documents, considering as case study thesis and dissertations from the Vidyanidhi Digital Library. This work resulted in an ontology for the agricultural domain, which was used to develop a system for retrieving agricultural information. The ontology was implemented in OWL, using the Protégé editor. AGROVOC was used to validate the ontology.

Xie *et al* [88] propose an architecture that applies the use of semantic Web services in a system that takes emergency agricultural decisions. The main goal of this system is to support rapid decisions, providing either multiple alternatives or partial solutions to emergency operations. The implemented system uses ontologies as base data model to allow data interpretation by machines in the Web. For this, it implemented three ontology repositories - respectively, cases, knowledge, and pre-planning repositories.

The work of [40] presents a framework for integrating heterogeneous ontologies in systems that extract knowledge about health care. The authors describe a process for generating ontologies in the health domain to optimize agricultural processes, considering production of more healthy food. This work uses AGROVOC as the agricultural terms provider, giving support to an ontology layer.

2.6 Kinds of Annotations

There are several kinds of annotation, which vary according to their multimedia format. Blaser [6] comments the use of annotations in form of sketches for use in geographic information systems (GIS). These sketches are text or voice comments, which describe one or multiple entities, a relationship between these entities, or the context of the thing/object being annotated. These annotations are used to describe characteristics that are hard or impossible to be graphically formulated, for instance, an address or the name of a building.

Suwa and Tversky [66] describe the use of sketches between architects using text and diagrams for annotating ideas and using them to verify new relationships and characteristics. New information suggests means for refining the ideas of the architects, thus creating a cyclical process (sketching, inspecting, and revising). Electronic devices like tablets and PDAs provide text annotation using free-hand sketches, as can be seen in [62, 69]. The following sections present some tools that provide annotations in different multimedia formats, and tools based on Semantic Web technologies.

2.6.1 Annotation Tools that are Based on Devices

bibPhone [45] is a prototype for attaching audio annotations in books. To create annotations, the (bibPhone) gadget is used. It is composed of a speaker and a microphone. The user can listen to information about a book as well as create an annotation about it. bibPhone also has an RFID [64] reader that identifies a book and sends its ID via BlueTooth to a PC. This PC searches by this book in a database. If there are sounds related to the book, the PC sends them via BlueTooth to the speaker of bibPhone. If no sounds are found, bibPhone alerts the user that the book is “empty” and recommends him to press the record button to attach an annotation about the book. Figure 2.1 shows bibPhone being used to listen annotations about a book. In addition to the gadget, the PC has an interface where the librarian can configure parameters like number of annotations to be listened per book when a user performs a search and whether bibPhone is in read or read/write mode.

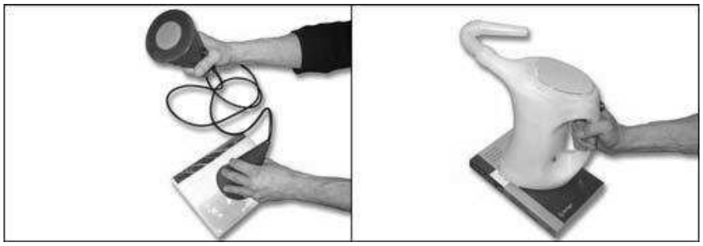


Figure 2.1: bibPhone gadgets in use - taken from [45]

Boom Chameleon [69] is a device that consists of a mobile monitor. It was created considering engineers that work in teams that have to discuss about projects. Boom Chameleon provides visualization of objects in 3D and creation of annotations using sounds, drawings, and sketches. When an object is shown, the user can annotate information about its visual perspective in various manners. A microphone allows capturing annotations from the user handling the device and discussions from the rest of the team. During the process of recording annotations, the visual perspective and the voice annotations are saved as different data flows.

Boom Chameleon's system also allows capture of gestures, divided in three categories: pen, flashlights, and pictures. The touch screen operation can work as a pen, thus allowing drawing in areas of the object, or in flashlight mode, in which a light ball appears in the area drawn by the user's fingers. Figure 2.2 shows Boom Chameleon being used to annotate parts of a vehicle. To start an annotation session, the user presses the record button and to capture vision, voice, and gesture flows. Thus, the user can get a 2D vision of the object (take a picture), draw a certain region of the image, and finally record the annotations about this region using speech.

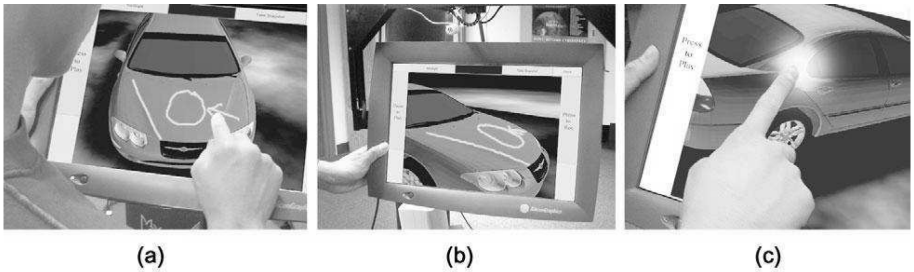


Figure 2.2: (a) Drawing with the pen; (b) Visualizing the annotation in other angle; (c) flashlight - taken from [69]

Digital Graffiti [11] is a tool for annotating the content of the Plasma Poster Network, allowing annotations in form of sketches and audio. The Plasma Poster Network [27] is a portal that allows users to post content and comment content from other users, using annotations. Digital Graffiti admits annotations from users using ordinary computers or PDAs.

Annotations can also be inserted offline, in personal devices, and subsequently uploaded in the portal. The annotations servlet allows annotations in form of sketches and audio. To annotate a digital content with sketches, the tool has a Java applet that uses the user's PDA stylus to insert a simple annotation. Through this applet, the user can annotate the desired content of the portal using the PDA. Figure 2.3 illustrates this process. Audio annotations are created using an embedded application that allows the user to record a brief comment with the microphone of the mobile device.

CommonSpace [56] is a commercial version of PREP Editor [10]. Its aim is to help teachers in classes and provide annotations for professional use, allowing text and audio annotations. Figure 2.4 shows the CommonSpace interface. The original text is shown in a column, whereas the annotations are shown in parallel columns so that they can be

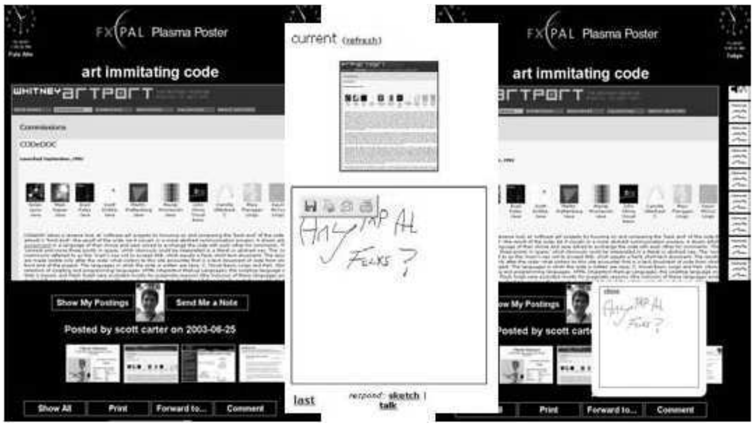
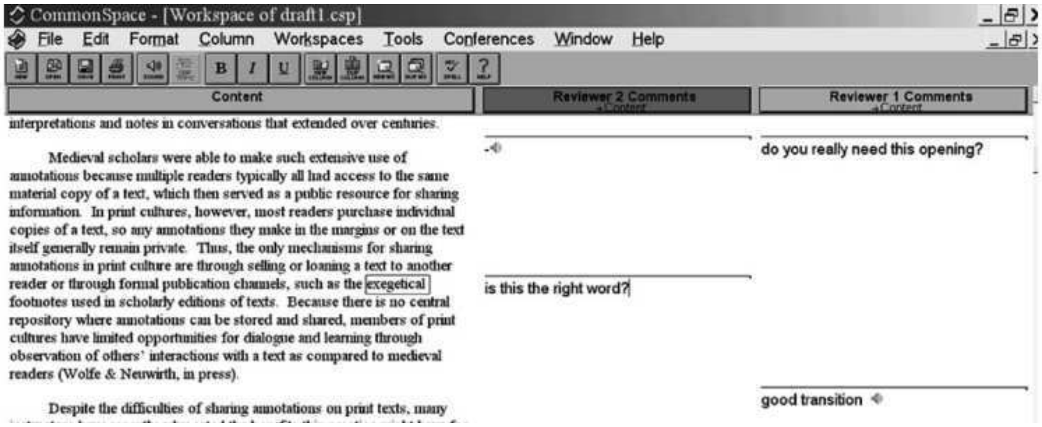


Figure 2.3: The desired content (left), the annotation being created (center), and the content with its new annotation (right) - taken from [11]

aligned to the corresponding text. The annotations of each reviser appear in separate columns. The user can create a library of often repeated annotations that may be stored for easy retrieval. The annotations are kept at the file being annotated.



interpretations and notes in conversations that extended over centuries.

Medieval scholars were able to make such extensive use of annotations because multiple readers typically all had access to the same material copy of a text, which then served as a public resource for sharing information. In print cultures, however, most readers purchase individual copies of a text, so any annotations they make in the margins or on the text itself generally remain private. Thus, the only mechanisms for sharing annotations in print culture are through selling or loaning a text to another reader or through formal publication channels, such as the exegetical footnotes used in scholarly editions of texts. Because there is no central repository where annotations can be stored and shared, members of print cultures have limited opportunities for dialogue and learning through observation of others' interactions with a text as compared to medieval readers (Wolfe & Neuwirth, in press).

Despite the difficulties of sharing annotations on print texts, many

do you really need this opening?

is this the right word?

good transition

Figure 2.4: Text annotation with CommonSpace - taken from [79]

XLibris [62] is a tablet monitor manufactured by Xerox that allows visualization of texts, imitating the sensation of reading printed books. It also allows the reader to visualize and search annotations of multiple documents simultaneously, using criteria as ink color or the timestamp when the annotation was created. The user can create annotations selecting sections of a text or directly annotating free-hand sketches in the screen using a stylus pen. Annotations are kept at the document being annotated. The main characteristic of XLibris is that it allows linking a single annotation to multiple text segments. Thus, a reader that wishes to show a link or a contradiction between different text parts can link these parts using a single annotation.

Space Pen [34] is part of the As If You Were Here system [18] and allows annotations of 3D images using text and sketches. This tool receives VRML [76] models posted by architects and converts them to Java 3D models so that they can be visualized in an ordinary internet browser. Thus, designers and members of the collaboration team can browse and annotate sketches over the surface of the models just like if these sketches were graffitis. The tool also allows textual annotations with tags in the Post-It [2] style, which can be attached and kept in the model.

eTrace [28] is a tool that allows annotations in 2D and 3D objects, using a pen. Figure 2.5 shows a 3D annotation using eTrace. The annotation in 3D objects is performed drawing or doodling over the object surface, so that the annotations also are in 3D. Thus, when the user performs some geometric transformation over the object (e.g; rotation, translation, or scale change), the annotations are transformed accordingly. The annotation is performed in a transparent window, moving around a 3D scene. The 2D and 3D annotations are independent of language and the scene representation format. Instead of editing the scene content, the annotation is codified, sent, and stored as a separate description; however, it is associated to the scene.

Sierra [54] is a prototype that allows image annotation and retrieval, applying the concept of superimposed information [49]. It uses CBISC [68], a framework which supports queries over image collections. Sierra is integrated to other components like Wordnet thesaurus [21] and SPARCE, a middleware for managing tags over text, audio, and video content [53]. The annotations are stored in a PostgreSQL database. Figure 2.6 shows an image of a fish being annotated using Sierra.

In the annotation process of Sierra, the user identifies an image, marks a region of interest, and annotates this region with keywords. Thus, the mark is created and all relevant information is stored. The content of the sub-image referenced by the mark is stored at CBISC and the annotations are stored in a database. In the retrieval process, the user identifies an image, marks a region inside the image and uses this image to query Sierra. It uses the sub-image referenced by the mark created by the user to query CBISC. Finally, Sierra retrieves a list of images or marks similar to the mark created by the user.

2.6.2 Annotation Tools and Approaches Based on Semantic Web Technologies

BOEMIE [25] (*Bootstrapping Ontology Evolution with Multimedia Information Extraction*) is an ontology-based tool for annotating text files and Web pages. This tool locates

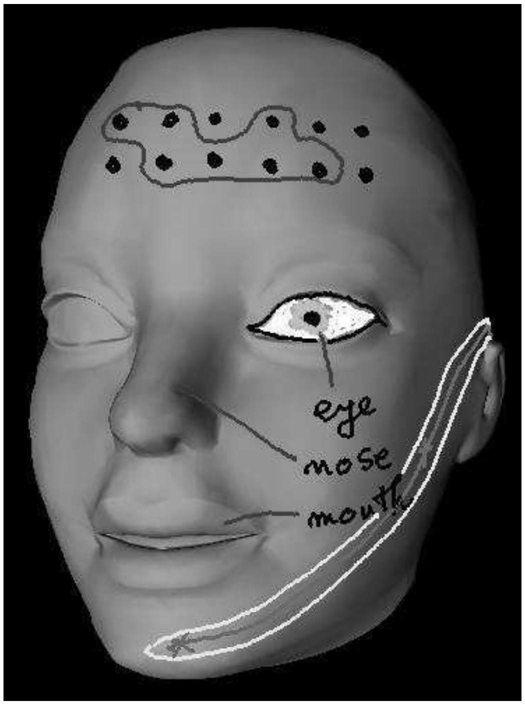
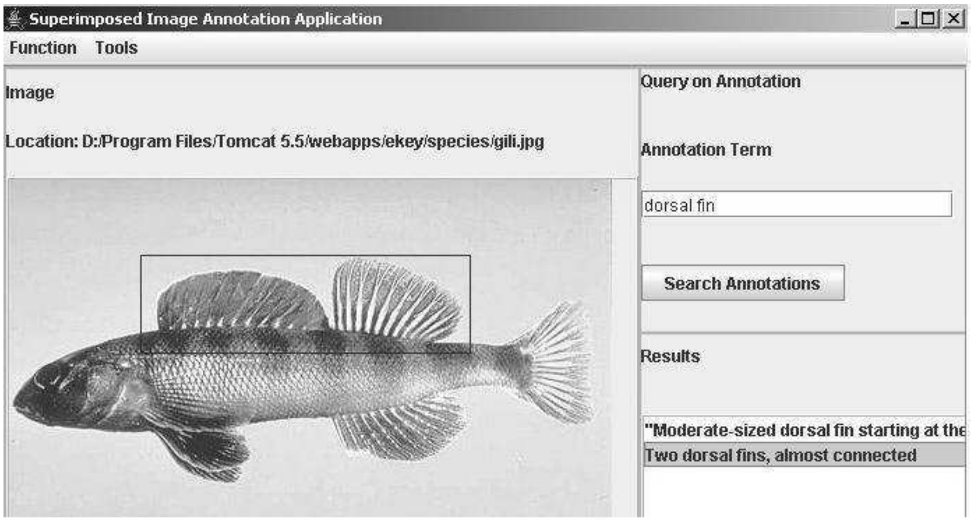


Figure 2.5: 3D annotation with eTrace - taken from [28]

blocks of text and assign them specific types, which are named entities from an ontology that uses the BOEMIE's terminology. The tool also provides annotation of relations between the named entities, grouping these relations in tables of specific types. Thus, named entities are considered as middle level concepts (MLC) of the ontology, whereas tables correspond to high level concepts (HLC). Furthermore, BOEMIE provides annotations in a higher level, annotating relations between HLC instances by creating linkages between tables. Figure 2.7 shows a screenshot of BOEMIE text annotation tool. The annotations created can be exported as OWL or XML files.

DocSS [8] (*Documentalist Support System*) is a Web-services based tool to generate ranked annotations about documents from the Netherlands Institute for Sound and Vision, which aims to facilitate the retrieval of such documents. The tool provides an environment in which documentalists can view and manipulate documents and metadata, receive annotation suggestions, create metadata, and search for semantically relevant documents within the collection. Figure 2.8 shows the search interface of DocSS. The annotation process was implemented as a Web Service, which is integrated to a Web interface. The process takes as input a text document and generates as output ranked annotation sug-



Location: D:/Program Files/Tomcat 5.5/webapps/ekey/species/gili.jpg

dorsal fin

"Moderate-sized dorsal fin starting at the

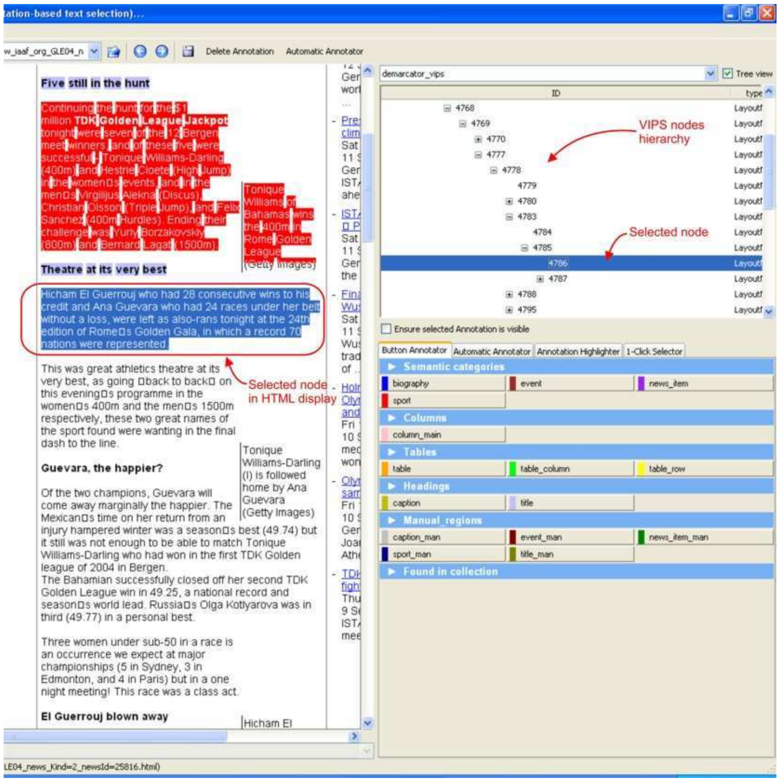
Two dorsal fins, almost connected

Figure 2.6: Fish annotated using Sierra - taken from [54]

gestions about the document. The user chooses the proper annotations and so order the system to store them. The annotations are represented in RDF and stored in a Sesame RDF repository [7].

Repp *et all* [63] proposes an approach for generating semantic annotations from multimedia objects. The work takes as case study university lectures, which are videos composed by slides that illustrate the lectures content, and voice that explains this content. Thus, the annotation process is divided in two phases. In the first, metadata are extracted from the content of the slides, using natural language techniques allied to a dictionary of synonyms. In the second phase, a synchronization between speech and slides is performed in order to extract metadata from the audio, so that an algorithm can be applied to get words from the voice stream that are similar to the words from the current slide page. Speech recognition techniques are applied for this. The generated metadata is represented using description logics and OWL ontologies.

In [75] is presented an architecture for collaborative semantic and pragmatic tagging among government agencies. In this environment, each agency has its data repository, where authorized people access restrict documents and data. Since the repositories are locally located, such people have wide knowledge about the content of these repositories. Thus, the work considers this fact to provide a system where people at their agencies create tags about documents to facilitate their retrieval by people from other agencies. Creation of tags are supported by ontologies and taxonomies, so that tags can be created using controlled vocabularies and be commonly shared and understood in different



Five still in the hunt

Continuing the hunt for the $1 million **TDK Golden League Jackpot** tonight were seven of the 12 Bergen meet winners, and of these five were successful - Tonique Williams-Darling (400m) and Hestrie Cloete (High Jump) in the women□s events, and in the men□s Virgilijus Alekna (Discus), Christian Olsson (Triple Jump) and Felix Sanchez (400m Hurdles). Ending their challenge was Yuriy Borzakovskiy (800m) and Bernard Lagat (1500m).

Tonique Williams of Bahamas wins the 400m in Rome Golden League (Getty Images)

Theatre at its very best

Hicham El Guerrouj who had 28 consecutive wins to his credit and Ana Guevara who had 24 races under her belt without a loss, were left as also-rans tonight at the 24th edition of Rome□s Golden Gala, in which a record 70 nations were represented.

This was great athletics theatre at its very best, as going □back to back□ on this evening□s programme in the women□s 400m and the men□s 1500m respectively, these two great names of the sport found were wanting in the final dash to the line.

Tonique Williams-Darling (l) is followed home by Ana Guevara (Getty Images)

Guevara, the happier?

Of the two champions, Guevara will come away marginally the happier. The Mexican□s time on her return from an injury hampered winter was a season□s best (49.74) but it still was not enough to be able to match Tonique Williams-Darling who had won in the first TDK Golden league of 2004 in Bergen.
The Bahamian successfully closed off her second TDK Golden League win in 49.25, a national record and season□s world lead. Russia□s Olga Kotlyarova was in third (49.77) in a personal best.

Three women under sub-50 in a race is an occurrence we expect at major championships (5 in Sydney, 3 in Edmonton, and 4 in Paris) but in a one night meeting! This race was a class act.

El Guerrouj blown away

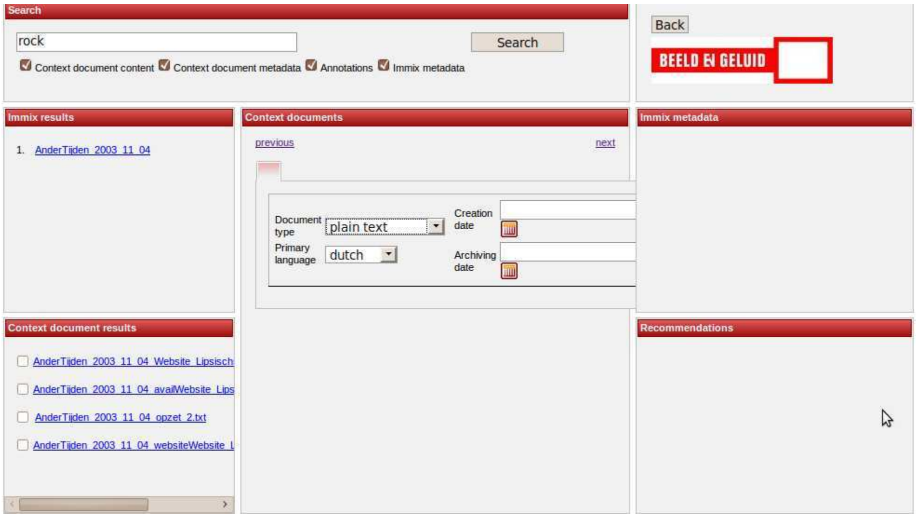
Hicham El

LE04_news_Kind=2_newsId=25816.html)

Figure 2.7: BOEMIE text annotation tool - extracted from [25]

environments. Furthermore, the system suggests to the user personal tags, which can be meaningful for him. The tags created by users are stored at local tag repositories which are hosted at each government agency.

Annotea [35] is a project developed by the World Wide Web Consortium (W3C) that intends to provide shared annotations of Web pages. The project involves a set of annotation servers, and annotation clients like Amaya [87] and Annozilla and add-ons like Annozilla [4], annoChump [80], and Janno [44]. Figure 2.9 shows an annotation being created via Amaya. The user can create annotations about the full page or parts thereof. Annotea provides annotations using comment, notes, and explanations in form of text. When an annotation is created, it is transformed in RDF code and sent to one of the Annotea's servers. Thus, an user can share his annotations with other users, change his annotations, or even reply one of his annotations or annotations from other users.



rock

☑ Context document content ☑ Context document metadata ☑ Annotations ☑ Immix metadata

1. AnderTijden 2003 11 04

Document type: plain text

Primary language: dutch

□ AnderTijden 2003 11 04 Website Lipsisch

□ AnderTijden 2003 11 04 availWebsite Lips

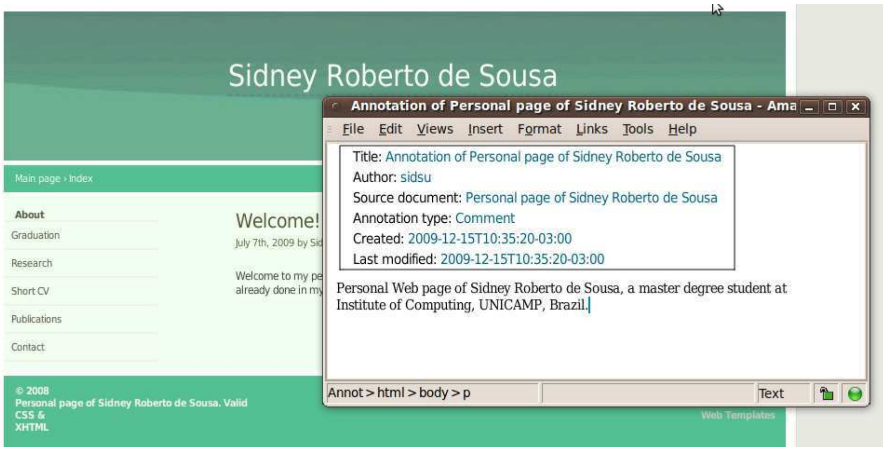
□ AnderTijden 2003 11 04 opzet 2.txt

□ AnderTijden 2003 11 04 websiteWebsite L

Figure 2.8: DocSS search interface

2.6.3 Analysis of the Studied Tools

The tools presented in the previous sections allow annotations in several formats and have different input methods (e.g.; from keyboard, touch screen, PDA, etc.). Part of these tools are semantics compliant, using technologies from the Semantic Web. They allow annotation of different types of data, like images, texts, and Web pages. Some of them were developed for specific users; however, their techniques show a broad scenario of how users can annotate data in different manners. Furthermore, while some of these tools store their annotations in databases, other tools keep the annotations inside of the annotated document itself. Table 2.1 shows a comparison of the tools studied, considering the type of the annotated data, the format of the annotations, whether they use Semantic Web (SW) technologies or not, how the annotations are stored, and the input method of the annotations by the users.



Title: Annotation of Personal page of Sidney Roberto de Sousa
Author: sidsu
Source document: Personal page of Sidney Roberto de Sousa
Annotation type: Comment
Created: 2009-12-15T10:35:20-03:00
Last modified: 2009-12-15T10:35:20-03:00

Personal Web page of Sidney Roberto de Sousa, a master degree student at Institute of Computing, UNICAMP, Brazil.

Figure 2.9: Using Annotea via Amaya

Tool	**Annotated Data**	**Format**	**SW Technologies**	**Storage**	**Input Method**
bibPhone	books	voice	-	database	microphone from the gadget
Boom Chameleon	3D images	sounds, drawings and sketches	-	different files for each kind of annotation	touch screen device
Digital Graffite	posts at the Plasma Posters Network portal	sketches and voice	-	portal's database	PDA (using stylus pen), keyboard, and mouse
CommonSpace	text	text and voice	-	local file	keyboard and microphone
XLibris	text	highlight and sketches	-	local file	pen
Space Pen	3D images	text and sketches	-	file	keyboard and mouse
eTrace	2D and 3D documents	sketches	-	separate file	pen
Sierra	2D images	text and marks	-	database	keyboard and mouse
BOEMIE	text, Web pages	ontologies	OWL ontologies	OWL or XML file	keyboard and mouse
DocSS	text documents	metadata fields	RDF	Sesame RDF database	keyboard and mouse
Repp *et all* [63]	video	ontologies	OWL ontologies	OWL file	keyboard and mouse
Warner and Chun [75]	multimedia documents	tags	Ontologies and taxonomies	Tags repositories	keyboard and mouse
Annotea	Web pages	comments and notes	RDF	RDF code at a server	keyboard and mouse

Table 2.1: Comparison of the studied tools

2.7 Conclusions

This chapter presented the basic concepts behind this dissertation, and related works. It also presented a survey on different annotation tools. This survey was necessary to understand how and why people create annotations. The next chapter presents the architecture implemented in this dissertation, containing the algorithms and methods used.

Chapter 3

An Architecture to Manage Semantic Annotations

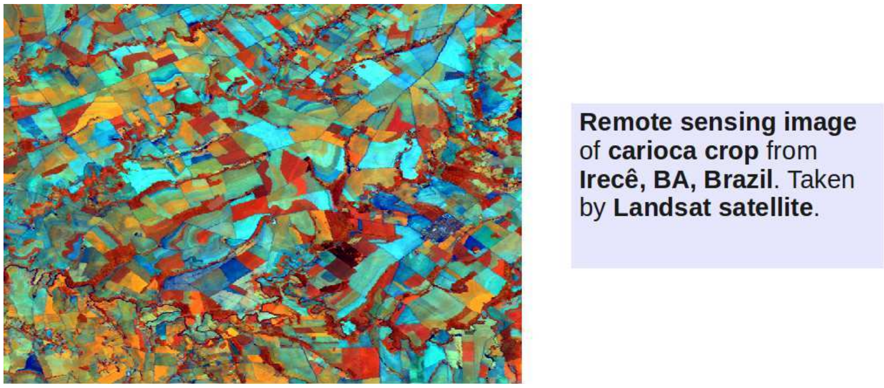
This chapter describes the service-based architecture for management of semantic annotations proposed in this dissertation. Section 3.1 presents the context of the work, giving an overview of the problem. Section 3.2 outlines the architecture, pointing out the services for managing semantic annotations. This architecture is discussed in the subsequent sections. Section 3.3 explains the configurations needed for importing, storing, and indexing ontology classes. Section 3.4 describes the process of semantic enrichment, which transforms an annotation into a semantic annotation. Section 3.5 discusses storage solutions for RDF documents. Finally, 3.6 presents the conclusions about this chapter.

3.1 Context of the Work - Running Example

Geospatial data are often created, used, and reused by specialists to produce useful information. In a typical scenario, experts find some resource of interest on the Web or in some repository and use it to produce other artifacts or even recommend the resource to other specialists. In order to reuse this resource, experts can make annotations that aim to describe the main characteristics that they singled out for the resource.

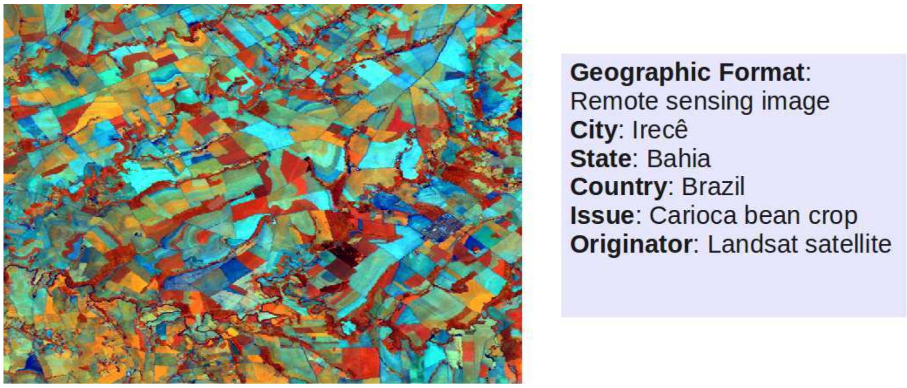
We now present a running example for the rest of the dissertation. Figure 3.1 shows an example of an annotation made by a specialist concerning a remote sensing image, where meta-information entities were highlighted. Here, the annotation indicates that it is a satellite image and contains patches of carioca beans crop.

To help annotation management, the specialist can structure it according to a metadata standard, separating each entity and assigning it to a metadata field. Figure 3.2 illustrates the same annotation from figure 3.1, using the FGDC geographic standard [20].



Remote sensing image of **carioca crop** from **Irecê, BA, Brazil**. Taken by **Landsat satellite**.

Figure 3.1: Simple free text annotation.



Geographic Format: Remote sensing image
City: Irecê
State: Bahia
Country: Brazil
Issue: Carioca bean crop
Originator: Landsat satellite

Figure 3.2: Annotation structured by metadata fields.

Despite the structure and semantics that metadata can provide, the content of the natural language fields may not be able to avoid problems such as ambiguity and misunderstanding [38]. For instance, the term “carioca bean” in the field *issue* may be familiar to a Brazilian agricultural specialist - due to the fact that “carioca” is a kind of bean widely cultivated in Brazil - but may not be so popular for people from other countries or for those who have never heard from this kind of bean. The use of an appropriate ontology term, beyond providing unique/universal meaning, enhances the semantics of the metadata fields thanks to the hierarchical structure that ontologies provide, where a concept has relations to other more general concepts. Figure 3.3 illustrates an ontology about crops, where the concept *Carioca* is associated to more including concepts that explain that *“Carioca is a kind of bean, which in turn is a kind of grain and crop”*.

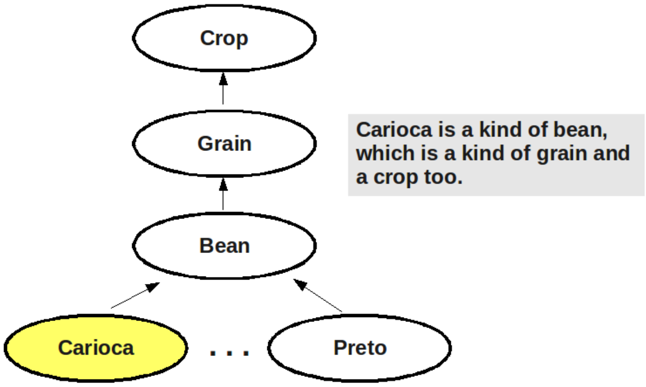

Crop
Grain
Carioca is a kind of bean, which is a kind of grain and a crop too.
Bean
Carioca
. . .
Preto

Figure 3.3: Ontology terms about beans.

Based on this approach and the definitions of [60, 47], we define annotations and semantic annotations:

Definition 1 *An annotation unit a is a triple* $< s, m, v >$*, where s is the subject being described, m the label of a metadata field, and v is a describing value assigned to m.*

Definition 2 *An annotation A is a set of one or more annotation units for the same subject.*

In this work, s is some multimedia data source from the Web (e.g., image, hypertext document, time series graph, etc.), m is a metadata field, and v is a natural language text provided by experts.

Definition 3 *A semantic annotation unit sa is a triple* $< s, m, o >$*, where s is the subject being described, m the same as definition 1, and o is an ontology term assigned to p, which semantically describes it.*

Definition 4 *A semantic annotation SA is a set of one or more semantic annotations units, having the same subject.*

Definition 5 *An annotation A (respectively, semantic annotation SA) is said to be structured according to a schema and content. The notion of schema is borrowed from database theory; here, it is an enumeration of metadata fields m to be filled with content v (respectively, o), and their domains.*

Thus, given a data source s, we have

$$a =< s, m_i, v_i >$$

$$sa =< s, am_i, o_i >$$

$$A = \{a_1, a_2, \ldots, a_n\}$$

$$SA = \{sa_1, sa_2, \ldots, sa_n\}$$

where the schema of the annotation = $\{m_i\}$.

Figure 3.4 shows the idea behind this dissertation - to create a software architecture that transforms an annotation A into a semantic annotation SA - i.e., to attack design and implementation issues in the box at the middle of the figure. Thus, the rest of this chapter discusses the software architecture and its algorithms. Metadata field values v_i are transformed into ontology terms o_i, thereby letting annotations become machine-processable.

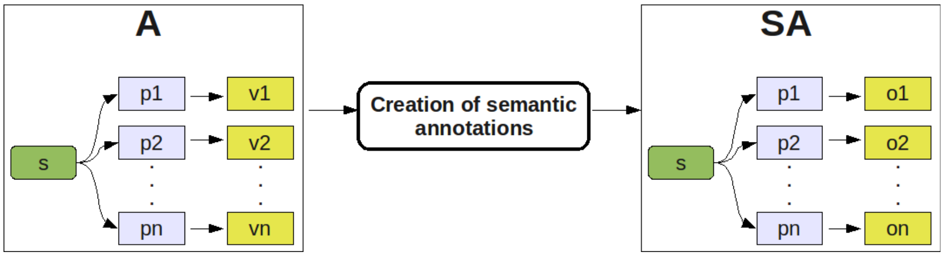


Figure 3.4: Creation of semantic annotations.

3.2 Architecture

Figure 3.5 shows the proposed architecture. It is composed by 3 layers. The Persistence layer provides to the Service layer access to the repositories containing ontology terms and semantic annotations. The Service layer, in turn, provides all services needed for the management of semantic annotations. These services can be accessed through the Interface layer. Arrows correspond to data flow (full edges) and service invocations and answers (dashed edges).

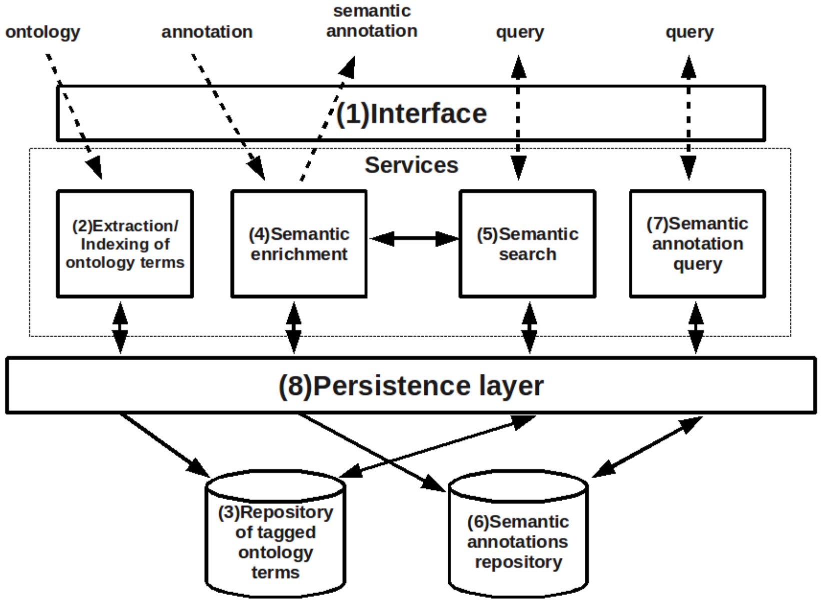

ontology
annotation
semantic annotation
query
query
(1)Interface
Services
(2)Extraction/ Indexing of ontology terms
(4)Semantic enrichment
(5)Semantic search
(7)Semantic annotation query
(8)Persistence layer
(3)Repository of tagged ontology terms
(6)Semantic annotations repository

Figure 3.5: Proposed architecture.

To better explain this architecture, we will take into account the typical usage scenario of its services and repositories. Before an annotation can be transformed into a semantic annotation, it is necessary to populate the repository with ontology terms (3) to support creation of semantic annotation units. For this, an expert will use the interface (1) to provide ontology URLs to the service of extraction and indexing of ontology terms (2). This service, given an ontology URL, downloads the ontology and extracts its terms, creates tags for them and stores terms and tags into the ontology terms repository (3). Tags are used to retrieve the terms of interest. Experts can include as many ontologies as needed. Section 3.3 presents the algorithm behind this process.

Once at least one ontology has been included, it is possible to use the semantic enrichment service (4) to transform an annotation A into a semantic annotation SA. In this scenario, the user provides an annotation to service (4). This service invokes the semantic search service (5), which accesses the ontology terms repository to get appropriate ontology terms. Each annotation unit a from A is transformed into a semantic annotation unit sa to compose SA, mapping the natural language content from a to an appropriate ontology term for sa. Section 3.4 presents the algorithms behind this process.

When a semantic annotation is finally created, it is stored into the semantic annotations repository (6). This repository can be accessed by experts through the interface, via the semantic annotation query service (7). Connection and access to ontology terms and semantic annotation repositories is provided by a persistence layer (8) (see more details in chapter 4).

3.3 Populating the Repository of Ontology Terms

Our model supports the transformation of annotation units into semantic annotation units. For this, we provide ways for the expert user to choose and insert appropriate domain ontologies. For each ontology the user wants to include, he/she just needs to inform the ontology's URL.

The extraction service starts by extracting all classes from the input ontology, to be used as terms for semantic annotation units. In order to provide retrieval of these terms during the process of creating a semantic annotation, each term receives a set of tags to semantically categorize it. Algorithm 1 corresponds to the extraction service. It receives as input an ontology and returns as output a set of tagged ontology terms. It starts by getting the URI of each ontology term (line 3). Tag assignment (lines 4) is performed as follows. Each class receives as tags the names of all its super classes, up to the ontology root (composite names are separated by blanks). It also receives as tags its synonyms. This way, each term becomes a pair $< URI, \{tags\} >$. Finally, the term is included in the terms set (line 5). The repository of ontology terms is in fact a repository of tagged terms.

Algorithm 1 Apply appropriate tags to ontology terms

```
Input: An ontology $O$
Output: A set of tagged ontology terms $T$
1: $T \leftarrow \emptyset$
2: for all class $c$ in $O$ do
3:     $u \leftarrow$ URI from $c$
4:     $tags \leftarrow superClassesNames(c) \cup getSynonyms(c)$
5:     $T \leftarrow T \cup < u, tags >$
6: end for
```

To better understand this process, consider the following example. Suppose that in the input ontology O the super classes of the *Bean* class are {*Grain, Crop, Raw, AgriculturalProduct*} and its synonyms[1] are {*bean plant, dome, noodle, attic, bonce,*

[1] in WordNet [21]

noggin}. So, applying algorithm 1 to *Bean* will result in the tagged term T_{Bean} = <*http://www.lis.ic.unicamp.br /˜sidney/agricZoning.owl#Bean,* {*bean, bean plant, dome, noodle, attic, bonce, noggin, grain, crop, raw, agricultural product*>}. However, applying algorithm 1 to *Carioca* will result in the tagged term $T_{Carioca}$ = <*http://www.lis.ic.unicamp .br/˜sidney/ agricZoning.owl#Carioca,* {*carioca, bean, grain, crop, raw, agricultural, product*>}, because no synonyms were found for *Carioca*.

Once all ontology classes are extracted and transformed into tagged terms according to algorithm 1, the search for these terms can be made over their tags.

3.4 Semantic Enrichment

Once at least one ontology is inserted, it is possible to execute the transformation process of an annotation into a semantic annotation. Algorithm 2 returns a ranked list of ontology terms $OntTerms$ that match the natural language text v from an annotation unit $< s, p, v >$.

Algorithm 2 Semantic search

```
Input: A natural language text $v$ from an annotation unit
Output: A set of appropriate ontology terms $OntTerms$
1: $hits \leftarrow search(v)$
2: if $hits \neq \emptyset$ then
3:     $OntTerms \leftarrow semanticRanking(hits)$
4:     $OntTerms \leftarrow syntacticRanking(OntTerms, v)$
5: else
6:     $OntTerms \leftarrow \emptyset$
7: end if
```

The output of algorithm 2, $OntTerms$, contains only ontology terms whose tags occur in v. All of these terms have some semantic relationship with v, due to the fact that their tags either match the words from v or are super classes or synonyms of those. Line 1 invokes the search method that returns a list of ontology terms that appear in v. For instance, if $v = Bean$ then $hits = \{T_{Bean}, T_{Carioca}\}$, because the string “bean” appears in the first two tags of T_{Bean} and the third tag of $T_{Carioca}$, but T_{Grain} will not be contained in *hits* because $tags_{T_{Grain}}$ = {*grain, food grain, cereal, crop, agricultural, product*}.

If the list of terms is not empty, its terms are firstly ranked as follows (line 3). Let t be a tagged term from $OntTerms$, occ_{t_i} the number of occurrences of words from v that appear in the tags from t_i, and N the number of elements at $OntTerms$; $OntTerms$ is sorted in a way so $occ_{t_1} \geq occ_{t_2} \geq occ_{t_3}, \ldots, occ_{t_{N-1}} \geq occ_{t_N}$. Figure 3.6 shows an example where terms are ranked according to the occurrence (occ_{t_i}) of $v = bean$ in their tag sets. T_{Grain} does not appear in $OntTerms$ because "bean" is not within T_{Grain} tags.

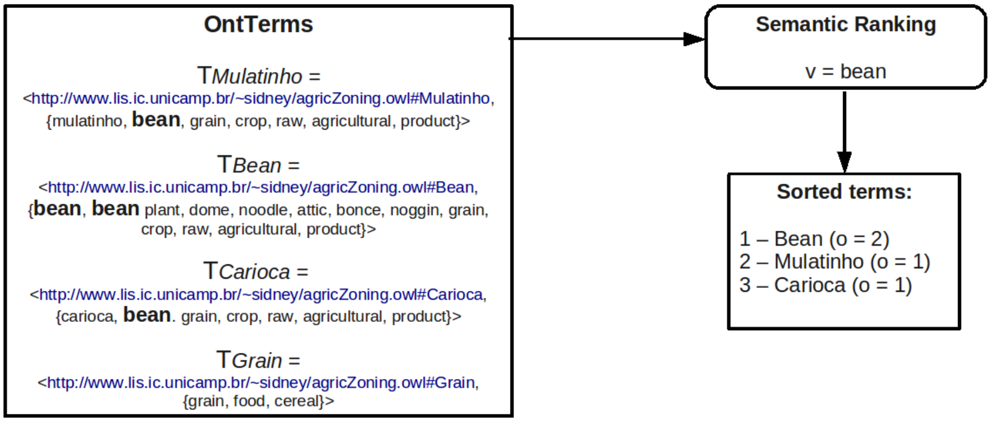

OntTerms
T*Mulatinho* =
<http://www.lis.ic.unicamp.br/~sidney/agricZoning.owl#Mulatinho,
{mulatinho, **bean**, grain, crop, raw, agricultural, product}>
T*Bean* =
<http://www.lis.ic.unicamp.br/~sidney/agricZoning.owl#Bean,
{**bean**, **bean** plant, dome, noodle, attic, bonce, noggin, grain, crop, raw, agricultural, product}>
T*Carioca* =
<http://www.lis.ic.unicamp.br/~sidney/agricZoning.owl#Carioca,
{carioca, **bean**. grain, crop, raw, agricultural, product}>
T*Grain* =
<http://www.lis.ic.unicamp.br/~sidney/agricZoning.owl#Grain,
{grain, food, cereal}>
Semantic Ranking
v = bean
Sorted terms:
1 – Bean (o = 2)
2 – Mulatinho (o = 1)
3 – Carioca (o = 1)

Figure 3.6: Example of semantic ranking.

Once all terms from T are semantically associated to v, a syntactic ranking is performed (line 4), in order to get the more relevant terms. For instance, if $v = \{crop\}$ and $OntTerms = \{Bean,\ AgriculturalCrop,\ Crop,\ Fruit\}$, after the syntactic ranking $OntTerms$ could change to $\{Crop,\ AgriculturalCrop,\ Bean,\ Fruit\}$.

Algorithm 3 shows the pseudo code of function *syntacticRanking(T,S)*. A score value is defined for each term t at the main loop (lines 2 to 20) according to its similarity to each word w from v, ignoring upper and lower cases. A score value can increase according to the following characteristics:

- **Similarity between t and w (line 5)**, which is calculated by *calculateSimilarity(t,w)* that implements the Jaro strings similarity function [33][2]. Return values vary from 0 to 1, where 0 is returned if the strings are totally different and 1 corresponds to exact matching. Given two strings $str_1 = a_1 \ldots a_K$ and $str_2 = b_1 \ldots b_L$, their similarity is given by

$$Jaro(str_1, str_2) = \frac{1}{3}\left(\frac{|str'_1|}{|str_1|} + \frac{|str'_2|}{|str_2|} + \frac{|str'_1| - Tr_{str'_1,str'_2}}{|str'_1|}\right)$$

where $str'_1 = a'_1 \ldots a'_K$ is the set of characters in str_1 which are common with str_2 in the same order that they appear in str_1, $str'_2 = b'_1 \ldots b'_L$ is the set of characters in str_2 which are common with str_1 in the same order that they appear in str_2,

[2] We decided to use Jaro function, since implementations of this function are easily found, and it has a good performance for short strings - for instance, ontology terms names.

and $Tr_{str'_1,str'_2}$ is half of the number of transpositions for str'_1 and str'_2, where a transposition for str'_1 and str'_2 is a position i such that $a'_i \neq b'_i$ [13]. For instance, let $str1$ be *crops* and $str2$ be *corps*, we have $str'_1 = crps$, $str'_2 = cops$, $|str1| = 5$, $|str2| = 5$, $|str'_1| = 4$, $|str'_2| = 4$, and $Tr_{str'_1,str'_2} = 0,5$. So, $Jaro(str_1, str_2) = 0,825$;

- **Exact matching (lines 6 and 7)**. If t exactly matches w, its score is increased by weight $we1$;
- **Occurrence of w at t (lines 9 and 10)**. If w occurs at t, score of t is increased by weight $we2$;
- **Common root word (lines 12 and 13)**. If t has the same root word as w, its score is increased by weight $we3$. The root is the primary lexical unit of a word. For instance, the words *federal* and *federation* have the same root word, *federa*. This was implemented using an external library, as will be seen in chapter 4.

In the implementation of the architecture, weights assigned were $we1 = 0.7$, $we2 = 0.5$, and $we3 = 0.3$. Once all scores are calculated, terms are sorted decreasingly according to their scores (lines 21 to 25). Thus, the first element of this list is used as default candidate ontology term to fill the corresponding metadata field of the semantic annotation unit. However, the user may choose another lower ranked term (see more details at chapter 4).

3.5 Storage Options for Semantic Annotations

From the moment that a semantic annotation is created, its valid RDF code is generated (box 4 of figure 3.5). At this point, it is necessary to persist this code for future use. RDF can be represented in various languages, some of which are more human-readable, like Notation3 (N3) [81], whereas others are more computer-processable, like RDF/XML [82]. Storage of RDF depends on the language used to represent it.

One possible solution for storing RDF is to use a native XML database to store and handle RDF/XML. Native XML databases use XQuery and XPath to query XML contents. These languages provide navigation over DOM trees[3] to retrieve full documents or specific parts of thereof.

[3] The XML DOM (Document Object Model) defines a standard way for accessing and manipulating documents compatible to XML, presenting them as a tree structure where elements, attributes, and text are nodes.

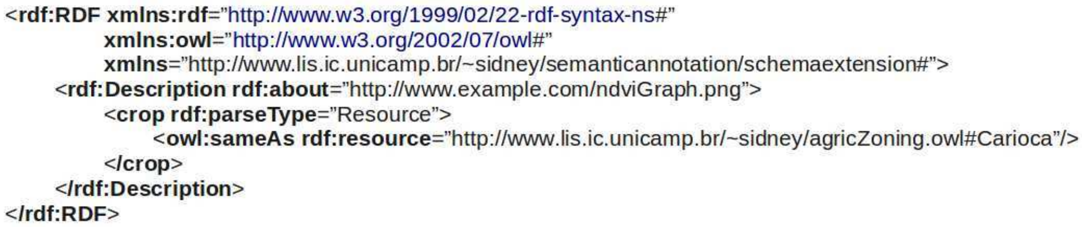
Algorithm 3 Syntactic ranking

Input: A set of ontology terms $OntTerms$, a natural language text v from an annotation unit

Output: A set of syntactically ranked ontology terms $OntTerms_1$

```
$i \leftarrow 1$
for all term $t$ in $OntTerms$ do
    $score[i] \leftarrow 0$
    for all word $w$ in $v$ do
        $score[i] \leftarrow score[i] + calculateSimilarity(t, w)$
        if $t = w$ then
            $score[i] = score[i] + we1$
        else
            if $t$ contains $w$ then
                $score[i] = score[i] + we2$
            else
                if $t$ has the same root word than $w$ then
                    $score[i] = score[i] + we3$
                end if
            end if
        end if
    end for
    $indexes[i] \leftarrow i$
    $i \leftarrow i + 1$
end for
$indexes \leftarrow sort(score, indexes)$
$OntTerms_1 \leftarrow \emptyset$
for all index $i$ from $indexes$, term $t$ from $OntTerms$ do
    $OntTerms_1 \leftarrow OntTerms_1 \cup term_i$
end for
```

For instance, consider the semantic annotation unit represented in RDF/XML, shown in figure 3.7. The *rdf:Description* element indicates a description of some resource from the Web. The *rdf:about* attribute identifies the resource (subject) using its URI. The *crop* element is a metadata field from the agricultural metadata extension used in this work, which refers to the crop to which the document refers. The *rdf:resource* attribute contains an ontology term and the *owl:sameAs* element indicates an "same as" relationship between the metadata field and this ontology term. Thus, this semantic annotation unit indicates that "the crop present by the document is a carioca bean crop" - the ontology term *http://www.lis.ic.unicamp.br/˜sidney/agricZoning.owl#Carioca*. If a user wants to retrieve this information, he could execute the XPath query */rdf:RDF/rdf:Description/ crop/owl:sameAs/@rdf:resource* and obtain the ontology term *"http://www.lis.ic.unicamp.br/ ˜sidney/agricZoning.owl#Carioca"* as result.



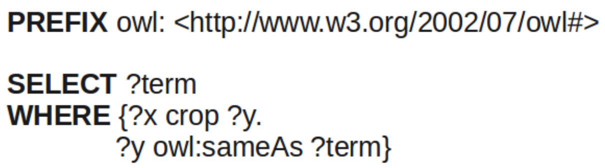
```
<rdf:RDF xmlns:rdf=”http://www.w3.org/1999/02/22-rdf-syntax-ns#”
         xmlns:owl=”http://www.w3.org/2002/07/owl#”
         xmlns=”http://www.lis.ic.unicamp.br/~sidney/semanticannotation/schemaextension#”>
    <rdf:Description rdf:about=”http://www.example.com/ndviGraph.png”>
         <crop rdf:parseType=”Resource”>
              <owl:sameAs rdf:resource=”http://www.lis.ic.unicamp.br/~sidney/agricZoning.owl#Carioca”/>
         </crop>
    </rdf:Description>
</rdf:RDF>
```

Figure 3.7: Example of semantic annotation unit.

There are some problems in using native XML databases for storing RDF. Firstly, queries over RDF are performed using the DOM tree structure from RDF/XML files, instead of RDF triples. Thus, the main purpose of using RDF is lost. Furthermore, some native XML databases, like eXist [19], do not allow storage of documents structured in XML-based languages like RDF/XML, OWL, and others, in contrast to some native XML databases like BaseX [70] and Sedna [51], which provide storage and handling of any content based on XML.

A better solution - which is the one chosen by this work - is the use of RDF databases, which are frameworks that provide CRUD (**C**reate, **R**eplace, **U**pdate, and **D**elete) operations over RDF content, beyond exportation of RDF in various languages and persistence into relational databases or binary files. In these frameworks, queries over RDF are performed using SPARQL [85], RDQL [83], or proprietary languages. Figure 3.8 shows a SPARQL query over the semantic annotation unit shown in figure 3.7. This query searches for a semantic annotation unit that contains as predicate the metadata field *crop*, and selects the ontology term *term*. Well known examples of such frameworks are Jena [78] and Sesame [7], which are distributed as APIs.

Still another solution is the use of relational databases, using SQL to query RDF.



```
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT ?term
WHERE {?x crop ?y.
          ?y owl:sameAs ?term}
```

Figure 3.8: Example of SPARQL query.

For this, a mapping from RDF to the entity-relationship (ER) model is needed. This solution is not as practical as the others presented in this section, since it has a higher implementation cost. Furthermore, for each different metadata schema used, a different mapping is needed. Hence, reuse of an ER model becomes very limited.

3.6 Conclusions

This chapter presented the architecture proposed to manage semantic annotations. It also presented the core algorithms within the architecture and discussed options to store annotations, showing why RDF databases were selected. The next chapter presents the architecture's implementation and an example of its use.

Chapter 4

Implementation Aspects

This chapter presents the implementation aspects of this dissertation. Section 4.1 presents a brief low-level description of the architecture proposed in section 3.2, considering implementation issues. Section 4.2 discusses the technologies used in the implementation of this dissertation. Section 4.3 presents an example of use of the implemented architecture. Section 4.4 presents the tests performed to validate the implemented services, using data from agricultural test cases. Finally, 4.5 presents the conclusions.

4.1 Technologies Used

Figure 4.1 reproduces figure 3.5, pointing out technology issues that will be discussed in this chapter. As seen in section 3.2, the architecture is composed by three layers. The Web Interface layer provides a set of Web pages from which users can access all the services provided by the Services layer.

The Services layer is composed by five services - numbered (2), (4), (5), (7), (8) in the figure. The service for extraction of ontology terms (2) receives as input the URL of an ontology available on the Web, extracts its terms, and stores them at the repository of tagged ontology terms (3). The service for creation of semantic annotation in RDF/XML (4) receives as input an annotation in XML and transforms each of its annotation units into semantic annotation units, choosing appropriate ontology terms for them. These terms are found via calls to the semantic search service (7). Alternatively, the user can create semantic annotations directly, invoking service (4) with appropriate ontology terms (see more details in section 4.2.2). As output, the service returns the corresponding RDF/XML semantic annotation for the input annotation.

Once a semantic annotation is created, it can be sent to the service for storage of RDF/XML (5) to be stored at the repository of semantic annotations (6). Users can retrieve semantic annotations accessing the service for querying semantic annotations (8).

SeRQL is a language for querying RDF that is integrating part of the framework Sesame [7]. This service receives a free text query, transforms it into a SeRQL query, and executes it at the repository of semantic annotations. The service returns all semantic annotations that match the query (see section 4.2.3 for more details).

The Persistence layer is basically composed by third-party software, provided as Java libraries. These libraries provide low-level connections and query operations over the data repositories. The management of the the repository of ontology terms (3) is performed by the Lucene API [67] (see section 4.2.1 for more details), whereas the repository of semantic annotations is managed by the Sesame API [7] (see section 4.2.3 for more details).

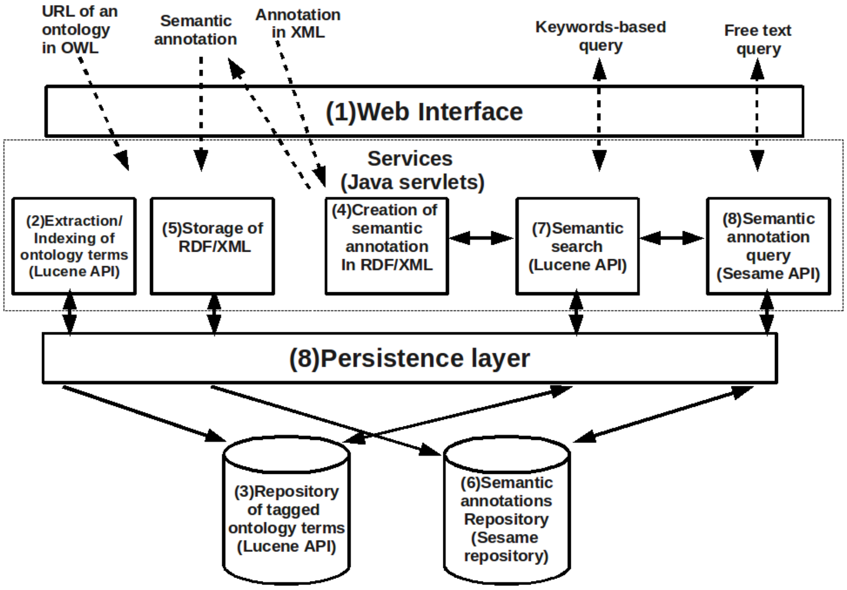

URL of an ontology in OWL
Semantic annotation
Annotation in XML
Keywords-based query
Free text query
(1)Web Interface
Services (Java servlets)
(2)Extraction/ Indexing of ontology terms (Lucene API)
(5)Storage of RDF/XML
(4)Creation of semantic annotation In RDF/XML
(7)Semantic search (Lucene API)
(8)Semantic annotation query (Sesame API)
(8)Persistence layer
(3)Repository of tagged ontology terms (Lucene API)
(6)Semantic annotations Repository (Sesame repository)

Figure 4.1: Proposed architecture of the implementation.

4.2 Implementation Details

The proposed architecture was implemented as a Web system. Implementing the architecture this way ensures that it can be used by geographically distributed clients, without the need for downloading executable softwares or repositories. Furthermore, it ensures that most data processing tasks can be done in the server and not in the user's machine.

The implementation comprises two kinds of Web technologies. The first one refers to languages and standards for development of applications on the Web. All services from figure 4.1 were implemented as separate Java servlets, which are classes that dynamically process HTTP requests and responses (i.e, service calls and processed data), adding dynamic content to a Web server using the Java platform [1]. Implementing the services as servlets allows loose coupling[1] and thus facilitates reuse of these services. Figure 4.2 illustrates the processing of servlet calls. The user at the client machine browses a Web page that provides access to a service (a servlet). This service is requested through an event inside the page (e.g., mouse click); some script behind the page - in general, JavaScript - sends an HTTP request to the server machine. This machine works as a Web server and hosts the requested servlet. The Web server sends the request to the servlet, which returns processed data to the Web server. Finally, the Web server sends an HTTP response containing the processed data to the client machine, and the user can see the data in a Web page.

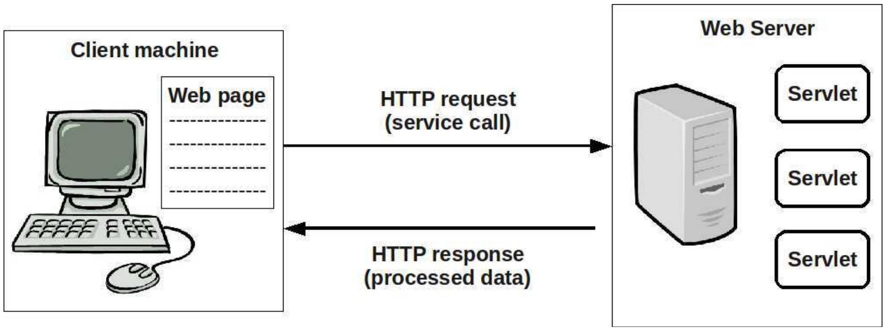


Figure 4.2: The servlets call processing.

In general, when the user requests some service via a Web page, the result is displayed by either refreshing the page or by loading another page. This can be a problem depending on the bandwidth of the network between the client and the server machine. To avoid this problem, the service calls in the implementation of our prototype are performed using AJAX (***A****synchronous* ***JA****vaScript and* ***X****ML*), a technology for creating faster and more interactive Web applications [74]. Figure 4.3 shows the communication between client and server using AJAX, exemplified using call to servlets. The idea behind this technology is to write JavaScript code to dynamically perform the HTTP requests to the Web server and receive the HTTP responses from it. Once the code receives the processed data from the servlet, these data are dynamically inserted at the current Web page that the user is browsing, avoiding refreshing or loading another page.

The second kind of Web technology involved in this dissertation refers to Semantic Web standards. In this work, RDF is used to represent semantic annotations. The

[1] That is, services do not depend (or have low dependency) on other services or software artifacts. It enables that services can be called as times as needed, independently of the application that call them.

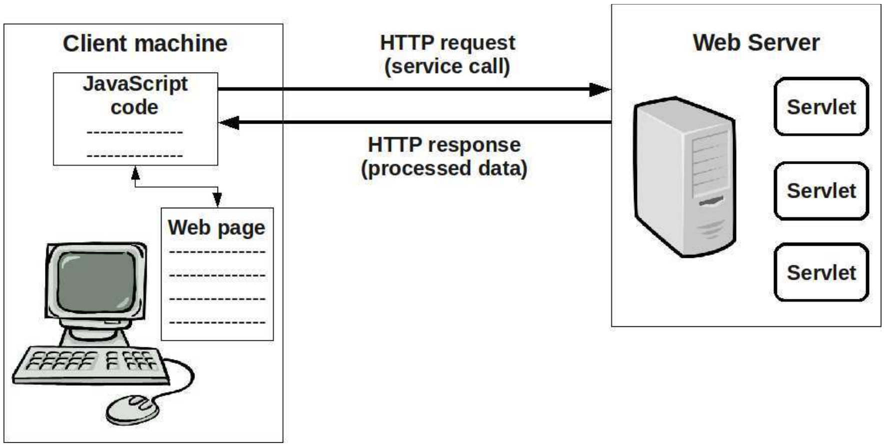


Figure 4.3: Communication between client and server using AJAX.

ontologies that are processed by the service of extraction and indexing of ontology terms are represented in OWL [86] (*recursive acronym for Web Ontology Language*), which is a language based on RDF. Furthermore, the service for querying semantic annotations uses the SeRQL.

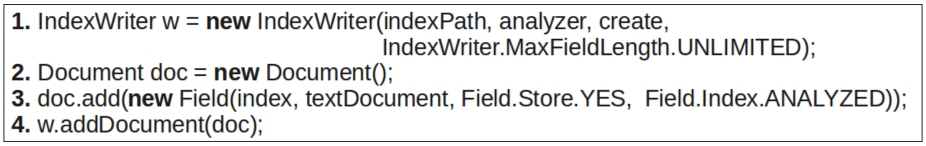
The following subsections explain how these technologies - associated to some other technologies - are applied in the services from figure 4.1. The algorithms behind these services are explained in chapter 3.

4.2.1 Management of Tagged Ontology Terms

As seen in section 3.3, each ontology term is associated with a set of tags, which are composed by the name of the term, its superclasses names, and its synonyms. The Jena API [78] was used to obtain the names of the superclasses, as well as to extract the ontology terms from the ontology inserted by the user. Jena is a framework for handling ontologies. Synonyms are obtained querying a PostgreSQL version of WordNet, a large lexical database of English, composed by nouns, verbs, adjectives, and adverbs [21].

In this work, a semantic annotation SA is a set of semantic annotation units. According to definition 3 in section 3.1, the ontology terms are used to semantically describe the semantic annotation units. Thus, it is necessary to provide a way for the user to choose appropriate ontology terms and also store the preferred ontologies. The first solution studied to attack this problem was to use Aondê [14], a Web service that provides various operations over ontologies. However, Aondê just provides searching for full ontologies, not terms.

Due to this problem, we implemented the service for extracting terms from an ontology (service 2 from figure 4.1) and the service to provide search for ontology terms (service 7). To implement these services, the Lucene API was used. Lucene is a text search engine, which provides indexing, storing, and searching of text-based documents. Figure 4.4 shows an example of usage of Lucene. An $IndexWriter$ object w is created at line 1. This object stores the indexed text documents in binary files. Its constructor receives four arguments: $indexPath$ is the path to the binary files where the documents are stored - the repository of indexed documents; *analyser* is the text analyser; *create* is a boolean value that is set to true if the repository must be created, or false if the repository already exists. Finally, $IndexWriter.MaxFieldLength$ sets the maximum length of the indexes - in this case, $UNLIMITED$.



```
1. IndexWriter w = new IndexWriter(indexPath, analyzer, create,
                                   IndexWriter.MaxFieldLength.UNLIMITED);
2. Document doc = new Document();
3. doc.add(new Field(index, textDocument, Field.Store.YES,  Field.Index.ANALYZED));
4. w.addDocument(doc);
```

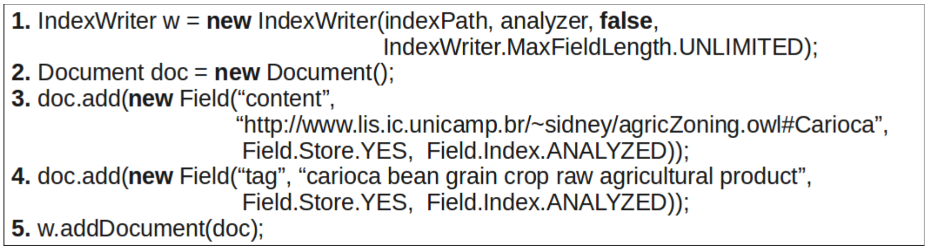
Figure 4.4: Example of usage of Lucene API.

At line 2 a document is created, and at line 3 a metadata field for this document is created. The *add* method receives as argument a $Field$ object, whose constructor receives four arguments: *index* is the name of the field; *textDocument* is the text value of the document; *Field.Store* indicates whether *textDocument* should be stored in the field - in this case, YES. Finally, *Field.Index* indicates whether the $Field$ object should be indexed, and if so, what should be done before indexing. In this case, $ANALYZED$ indicates that *textDocument* should be analyzed before indexing. At line 4, the document is stored in the repository.

In the repository of tagged ontology terms, each ontology term is associated with two metadata fields, *tag* and *content*. The *tag* field contains the set of tags of the ontology term, whereas the *content* field contains the URI of the term. Figure 4.5 shows how tagged ontology terms are stored and indexed by Lucene, taking as example the term $Carioca$. A *content* field is created at line 3, containing the URI of the $Carioca$ term, and a *tag* field is created at line 4, containing the tags associated to the term.

4.2.2 Creating Semantic Annotations

The architecture's goal is to manage semantic annotations. It supports creation of semantic annotations directly (the user interacts with service numbered 4 in figure 4.1) or, alternatively, by transformation of annotation units in semantic annotations units. Additionally, users can use the architecture to manually create annotations. The manual creation of annotations is as follows.



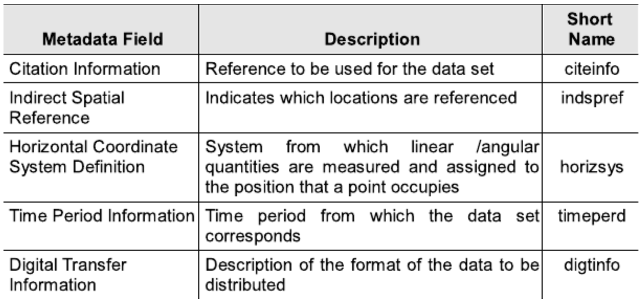
```
1. IndexWriter w = new IndexWriter(indexPath, analyzer, false,
                                        IndexWriter.MaxFieldLength.UNLIMITED);
2. Document doc = new Document();
3. doc.add(new Field(“content”,
                        “http://www.lis.ic.unicamp.br/~sidney/agricZoning.owl#Carioca”,
                         Field.Store.YES,  Field.Index.ANALYZED));
4. doc.add(new Field(“tag”, “carioca bean grain crop raw agricultural product”,
                         Field.Store.YES,  Field.Index.ANALYZED));
5. w.addDocument(doc);
```

Figure 4.5: Storage and indexing of a tagged ontology term on Lucene.

Once the user has chosen a resource to be annotated, he needs to choose a metadata set to be used. The implementation provides two different sets of metadata: pure FGDC metadata or FGDC metadata plus an agricultural extension. The use of each metadata field is optional.

The Federal Geographic Data Committee Metadata (FGDC Metadata) standard is an open standard which defines fields needed to catalog and publish geographic meta-information. It provides knowledge about the resources, indicating whether it meets the user's expectation, and where/how to find it. Use of a specific section or element is either mandatory or optional [12].

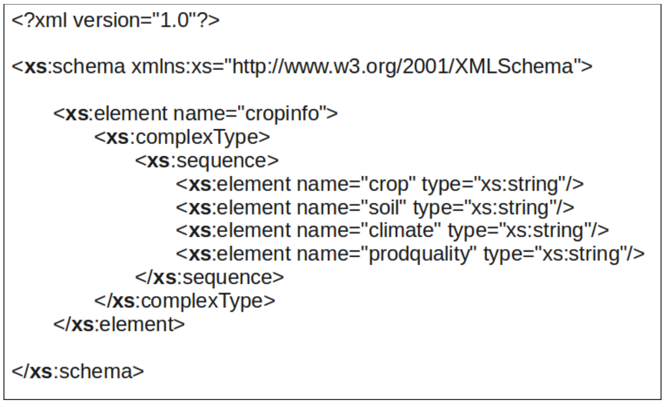
Due to the large number of elements from the FGDC standard and the non-mandatory use thereof, this work uses just a minimum set of elements needed to create concise semantic annotations. Figure 4.6 shows a table explaining the elements used. Each element shown in the table is composed by other specific elements, which were abstracted in the table. The first column shows the name of the elements. The second column shows a brief description of each element. Finally, the last column shows the short name of each element.



Metadata Field	Description	Short Name
Citation Information	Reference to be used for the data set	citeinfo
Indirect Spatial Reference	Indicates which locations are referenced	indspref
Horizontal Coordinate System Definition	System from which linear /angular quantities are measured and assigned to the position that a point occupies	horizsys
Time Period Information	Time period from which the data set corresponds	timeperd
Digital Transfer Information	Description of the format of the data to be distributed	digtinfo

Figure 4.6: Used elements from the FGDC metadata standard.

A short agricultural metadata extension was created in this work to cover some agricultural issues. Its use is optional, to be adopted only if some agricultural characteristic needs to be detected in the resource being annotated, for instance, a kind of crop or soil. Figure 4.7 shows the XML Schema of this extension. It contains just one session, called *Crop Information* (cropinfo), which comprises four elements. The *Crop* element means the kind of crop to which the annotated document refers; the *Soil* element refers to the kind of soil detected in the document; *Climate* means the kind of climate of the geographical region which the document refers to; and the *Production Quality* element (prodquality) is used to explain characteristics about quality of crop production.



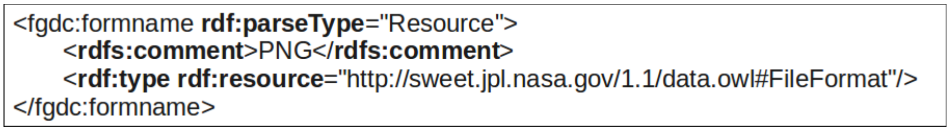
```
<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:element name="cropinfo">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="crop" type="xs:string"/>
                <xs:element name="soil" type="xs:string"/>
                <xs:element name="climate" type="xs:string"/>
                <xs:element name="prodquality" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

</xs:schema>
```

Figure 4.7: XML Schema of the agricultural metadata extension.

Once the schema is defined, the user can manually enter natural language expressions for each field, thereby creating annotations. Semantic annotations can be created when at least one ontology is inserted by the user and its terms are indexed and stored. The first requirement to create a semantic annotation is the URL of a valid resource on the Web (the subject of the semantic annotation units). Users can manually change an annotation unit into a semantic annotation unit by invoking service numbered 7 in figure 4.1, which returns a ranked list of ontology terms that match the natural language field.

Once the semantic annotation is created, its corresponding RDF/XML code is created. This mapping process is performed for each semantic annotation unit. Two different mappings are possible, depending on the kind of relationship between the natural language text and the ontology term that the user has specified.

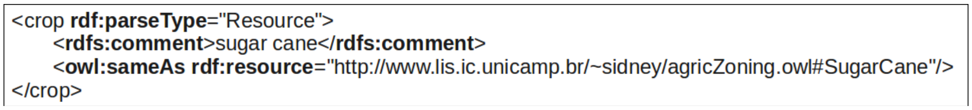
The first kind of mapping corresponds to the “is a” relationship. Figure 4.8 shows and adaptation of the RDF/XML code for this relationship. The *fgdc:formname* element from the FGDC standard refers to the name of the file format of the document being annotated. The *rdf:parseType="Resource"* attribute indicates that *fgdc:formname* contains other elements. The natural language text from the annotation unit is kept for human understanding and encapsulated by the *rdfs:comment* element, which is used to write natural language comments. The ontology term at *rdf:resource* is encapsulated by the *rdf:type* element, which indicates that "PNG *is a* type of file format", that is, a specialization.



```
<fgdc:formname rdf:parseType="Resource">
    <rdfs:comment>PNG</rdfs:comment>
    <rdf:type rdf:resource="http://sweet.jpl.nasa.gov/1.1/data.owl#FileFormat"/>
</fgdc:formname>
```

Figure 4.8: *Is a* relationship in RDF/XML.

The other kind of mapping corresponds to the "same as" relationship. Figure 4.9 shows the corresponding RDF/XML code for this relationship. In this case, the ontology term at *rdf:resource* is encapsulated by the *owl:sameAs* element, which indicates that the natural language text "sugar cane" means the same thing as the ontology term *SugarCane*.



```
<crop rdf:parseType="Resource">
    <rdfs:comment>sugar cane</rdfs:comment>
    <owl:sameAs rdf:resource="http://www.lis.ic.unicamp.br/~sidney/agricZoning.owl#SugarCane"/>
</crop>
```

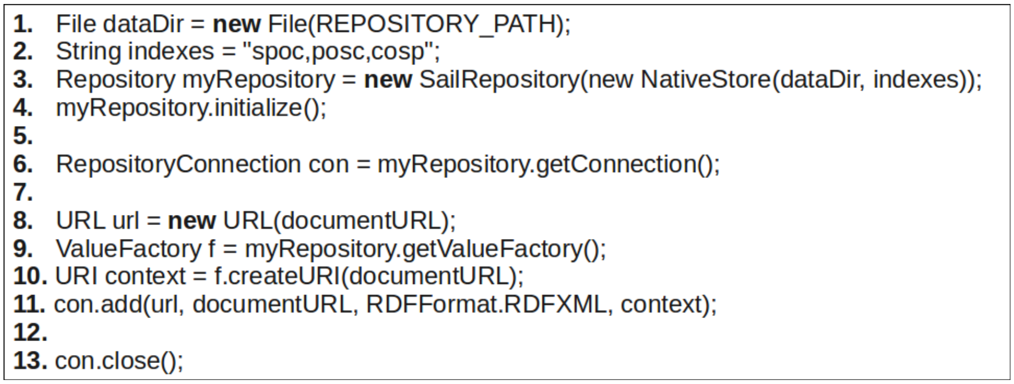
Figure 4.9: *Same as* relationship in RDF/XML.

4.2.3 The Repository of Semantic Annotations

When a semantic annotation is created, the corresponding RDF/XML code is generated and stored at the repository of semantic annotations. This repository was implemented using Sesame, a framework for storage and querying of RDF data [7]. It supports SeRQL and SPARQL querying and provides storage of RDF in binary files or relational databases (PostgreSQL and MySQL).

The repository consists of a set of binary files managed by Sesame, which provides methods for inserting RDF files in the repository. Figure 4.10 shows the code needed to insert an RDF/XML document in the repository. The first line specifies the directory where the binary files are saved. The second line contains the kind of indexes that will be created to optimize performance for query patterns that occur frequently. The letters *c*, *o*, *p*, and *s* respectively mean *context*, *object*, *predicate*, and *subject*. At lines 3 and 4 the repository is opened (or created, when it does not exist) and initialized.

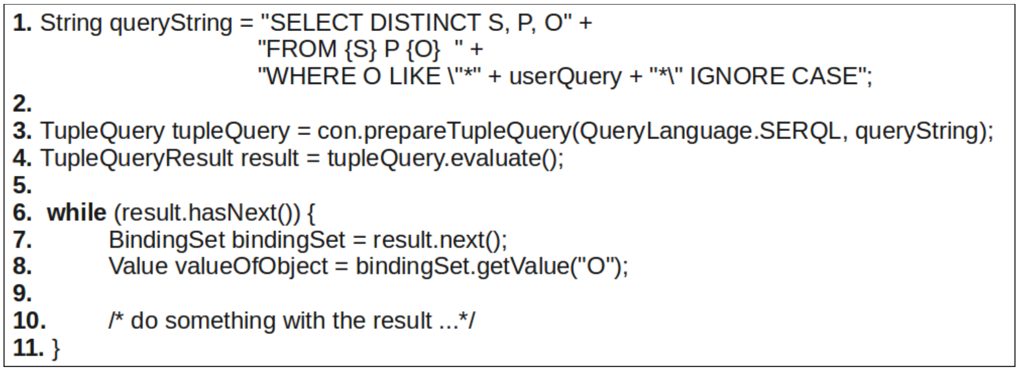
A connection to the repository is opened at line 6. At lines 8 to 10 a URI is created for the document to be stored. This URI plays the role of a primary key for the document. Finally, at line 11 the document is stored and the connection is closed at line 13.



```
File dataDir = new File(REPOSITORY_PATH);
String indexes = "spoc,posc,cosp";
Repository myRepository = new SailRepository(new NativeStore(dataDir, indexes));
myRepository.initialize();

RepositoryConnection con = myRepository.getConnection();

URL url = new URL(documentURL);
ValueFactory f = myRepository.getValueFactory();
URI context = f.createURI(documentURL);
con.add(url, documentURL, RDFFormat.RDFXML, context);

con.close();
```

Figure 4.10: Insertion of a RDF/XML document in the repository of semantic annotations.

Figure 4.11 shows the code needed to execute a SeRQL query over the repository of semantic annotations. The query string is created at line 1. It selects RDF triples $< subject, predicate, object >$ for which the object matches the user's query ($userQuery$). For instance, if $userQuery = corn$, the query selects the triples where either the ontology term or the natural language text contain "corn", ignoring upper and lower cases. Lines 3 and 4 execute the query. The triples returned by the query are processed inside the *while* loop at lines 6 to 11.



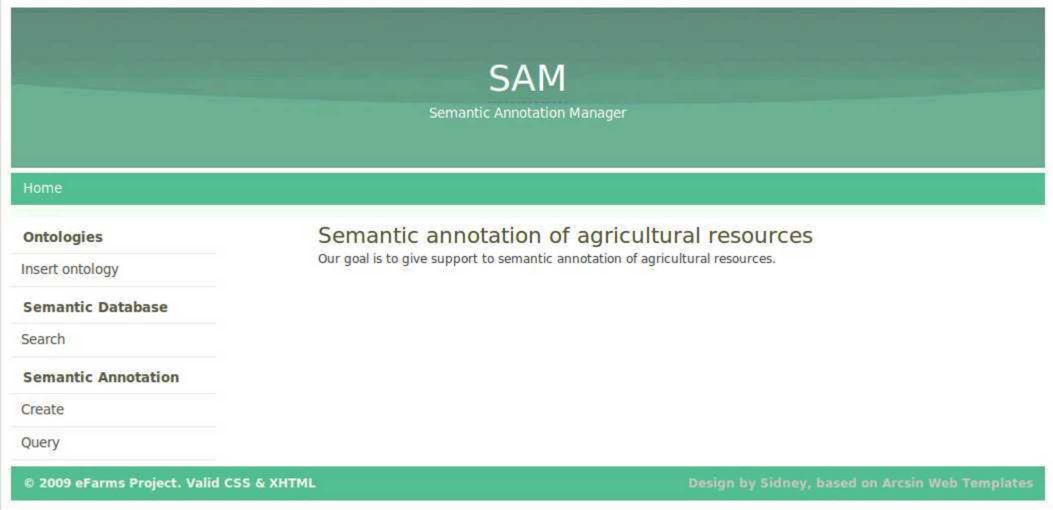
```
1. String queryString = "SELECT DISTINCT S, P, O" +
                        "FROM {S} P {O}  " +
                        "WHERE O LIKE \"*" + userQuery + "*\" IGNORE CASE";
2.
3. TupleQuery tupleQuery = con.prepareTupleQuery(QueryLanguage.SERQL, queryString);
4. TupleQueryResult result = tupleQuery.evaluate();
5.
6.  while (result.hasNext()) {
7.          BindingSet bindingSet = result.next();
8.          Value valueOfObject = bindingSet.getValue("O");
9.
10.         /* do something with the result ...*/
11. }
```

Figure 4.11: SeRQL query over the repository of semantic annotations.

A user query Q over the repository of semantic annotations is performed as follows. First, a query is sent to the semantic search service (service 7 in figure 4.1)), in order to find tagged ontology terms related to Q. For each ontology term returned, the service for querying semantic annotations (service 8 in figure 4.1) tries to find semantic annotation units - at the repository of semantic annotations - that contains this term; for each semantic annotation unit that contains the term, its corresponding semantic annotation is returned as result. If there are no ontology terms that match Q or there are no semantic annotation units which contains any of the ontology terms, the service performs Q directly over the repository of semantic annotations, searching over the natural language text of the *rdfs:comment* tags.

4.3 Example of Use

Figure 4.12 shows the main page of SAM (*Semantic Annotation Manager*), the implemented architecture of this dissertation. The menu on the left side is divided in three sub items. In the *Ontologies* item, the user request insertion of a new ontology in the option “Insert Ontology”. In *Semantic Database*, the user can query the repository of tagged ontology terms, using the option “Search”. This option was enabled in order to let the user know about the tagged terms which are stored at the repository, before creating semantic annotations. The last item (*Semantic Annotation*) contains options to create and query semantic annotations.


SAM

Semantic Annotation Manager

Semantic annotation of agricultural resources

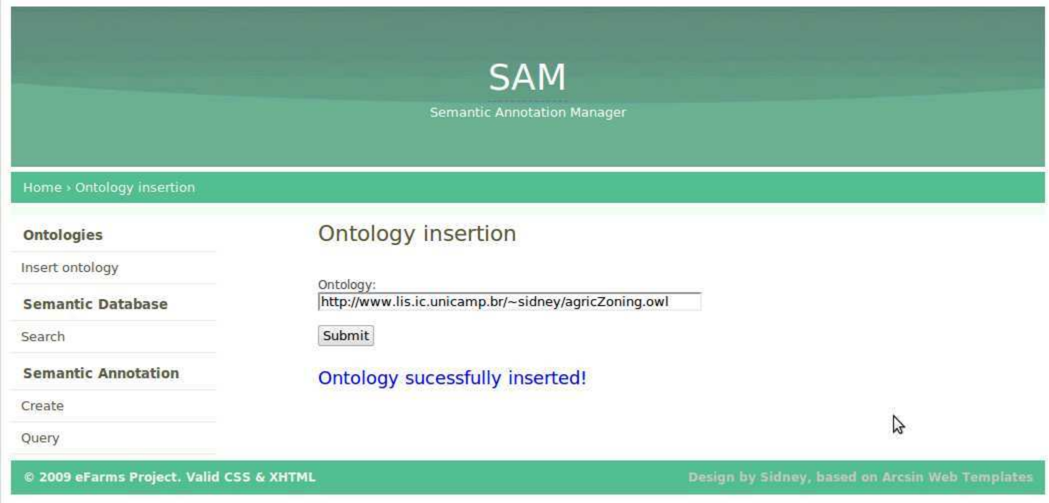
Our goal is to give support to semantic annotation of agricultural resources.

© 2009 eFarms Project. Valid CSS & XHTML

Design by Sidney, based on Arcsin Web Templates

Figure 4.12: SAM - Semantic Annotation Manager main page.

Let us go back to our running example, and consider that we want to create a semantic annotation for the remote sensing image from figure 3.2. The first step is to insert the ontologies needed. Figure 4.13 shows the page for insertion of ontologies. To insert an ontology, the user needs to provide the URL of the ontology and click on the *Submit* button. Here, AJAX was used so that the user is informed about the status of the insertion process without the need for reloading the page. This invokes the service that extracts and indexes ontology terms; this service may take some minutes to perform the processing needed, depending on the size of the ontology. The tagged ontology terms are saved at the repository, so the user does not have to repeat this process again. When the processing is ended, the user is informed with a “success” message.


SAM
Semantic Annotation Manager
Home › Ontology insertion
Ontologies
Insert ontology
Semantic Database
Search
Semantic Annotation
Create
Query
Submit
© 2009 eFarms Project. Valid CSS & XHTML
Design by Sidney, based on Arcsin Web Templates

Ontology insertion

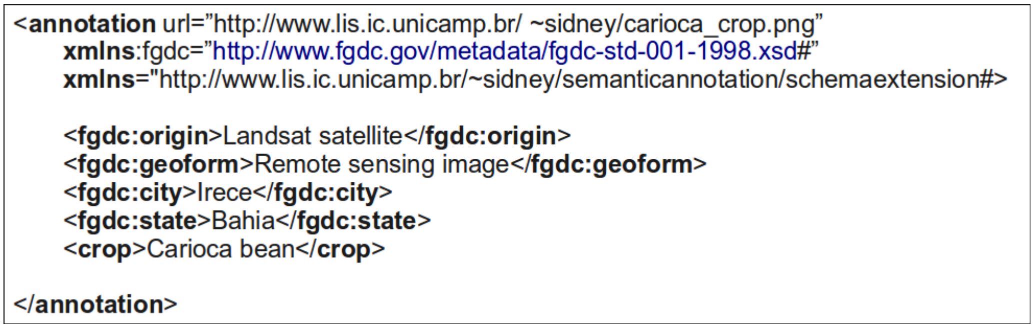
Ontology: http://www.lis.ic.unicamp.br/~sidney/agricZoning.owl

Ontology sucessfully inserted!

Figure 4.13: Inserting an ontology.

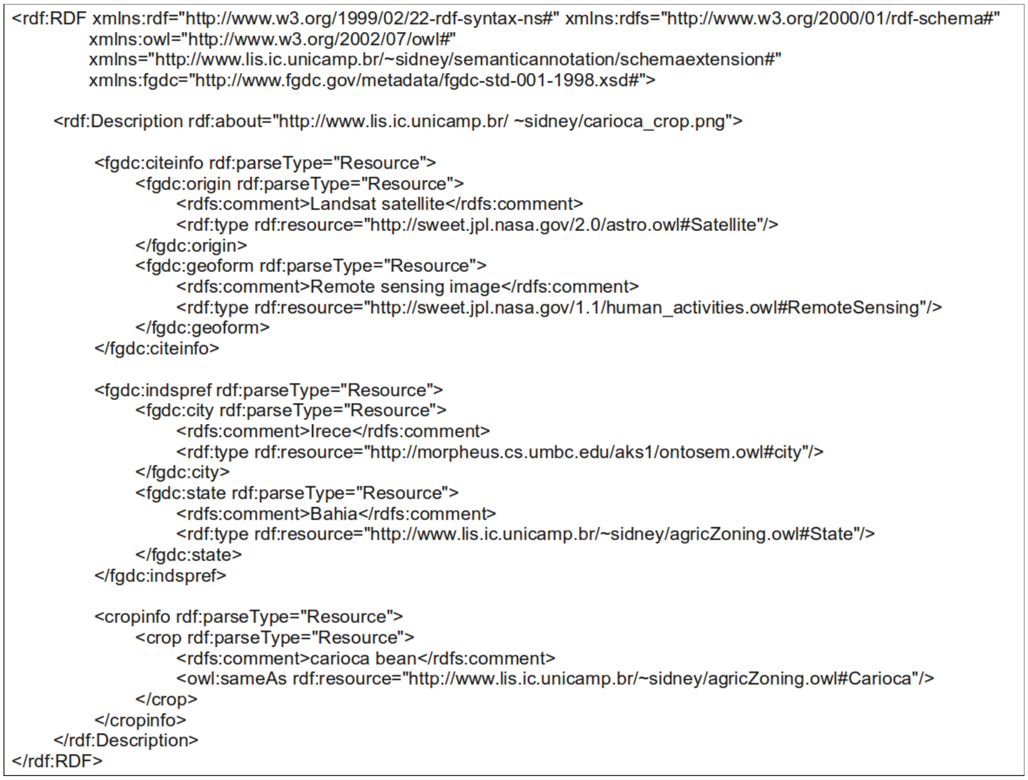
Semantic annotations can be created in two different ways. In the first, the user already has an annotation and just wants to transform it into a semantic annotation. Figure 4.14 shows an example of an input annotation. It contains metadata about the resource located at the URL specified in the *url* attribute. The service for creating semantic annotations receives this annotation and, for each annotation unit, chooses the first ontology term from the ranked terms returned by the semantic search service (numbered 7 in figure 4.1). Figure 4.15 shows the corresponding semantic annotation automatically generated for this annotation.

The second way is to use the SAM interface for creating semantic annotations. Figure 4.16 shows the page to request creation of semantic annotations. Once the ontology is inserted, the user can create these annotations directly o request transformation of annotations into semantic annotations. First, the user needs to inform the URL of the document and click on the *Validate* button so that the URL can be validated, that is, to confirm the existence of the document. The architecture does not allow creation of semantic annotations of non-existing documents. Once the existence of the document is confirmed, the user needs to choose a metadata schema - pure FGDC or FGDC plus the



```
<annotation url=”http://www.lis.ic.unicamp.br/ ~sidney/carioca_crop.png”
    xmlns:fgdc=”http://www.fgdc.gov/metadata/fgdc-std-001-1998.xsd#”
    xmlns="http://www.lis.ic.unicamp.br/~sidney/semanticannotation/schemaextension#>

    <fgdc:origin>Landsat satellite</fgdc:origin>
    <fgdc:geoform>Remote sensing image</fgdc:geoform>
    <fgdc:city>Irece</fgdc:city>
    <fgdc:state>Bahia</fgdc:state>
    <crop>Carioca bean</crop>

</annotation>
```

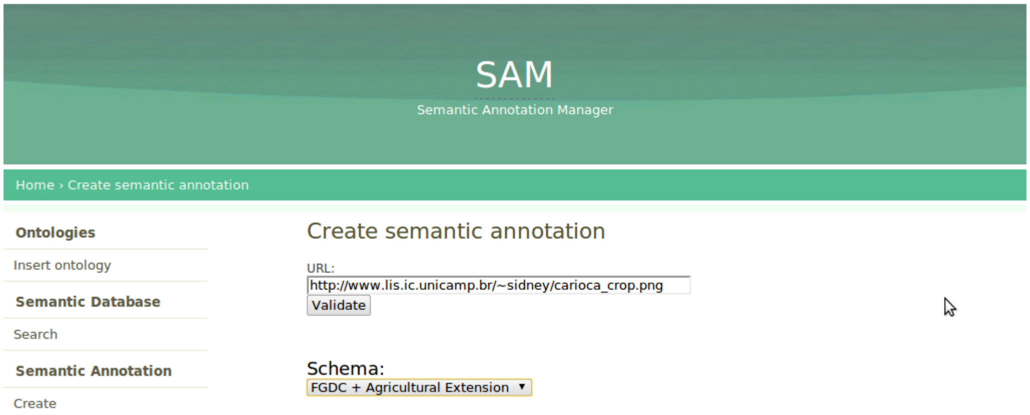
Figure 4.14: Input annotation XML for running example.



```
<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="http://www.lis.ic.unicamp.br/~sidney/semanticannotation/schemaextension#"
        xmlns:fgdc="http://www.fgdc.gov/metadata/fgdc-std-001-1998.xsd#">

    <rdf:Description rdf:about="http://www.lis.ic.unicamp.br/ ~sidney/carioca_crop.png">

        <fgdc:citeinfo rdf:parseType="Resource">
            <fgdc:origin rdf:parseType="Resource">
                <rdfs:comment>Landsat satellite</rdfs:comment>
                <rdf:type rdf:resource="http://sweet.jpl.nasa.gov/2.0/astro.owl#Satellite"/>
            </fgdc:origin>
            <fgdc:geoform rdf:parseType="Resource">
                <rdfs:comment>Remote sensing image</rdfs:comment>
                <rdf:type rdf:resource="http://sweet.jpl.nasa.gov/1.1/human_activities.owl#RemoteSensing"/>
            </fgdc:geoform>
        </fgdc:citeinfo>

        <fgdc:indspref rdf:parseType="Resource">
            <fgdc:city rdf:parseType="Resource">
                <rdfs:comment>Irece</rdfs:comment>
                <rdf:type rdf:resource="http://morpheus.cs.umbc.edu/aks1/ontosem.owl#city"/>
            </fgdc:city>
            <fgdc:state rdf:parseType="Resource">
                <rdfs:comment>Bahia</rdfs:comment>
                <rdf:type rdf:resource="http://www.lis.ic.unicamp.br/~sidney/agricZoning.owl#State"/>
            </fgdc:state>
        </fgdc:indspref>

        <cropinfo rdf:parseType="Resource">
            <crop rdf:parseType="Resource">
                <rdfs:comment>carioca bean</rdfs:comment>
                <owl:sameAs rdf:resource="http://www.lis.ic.unicamp.br/~sidney/agricZoning.owl#Carioca"/>
            </crop>
        </cropinfo>
    </rdf:Description>
</rdf:RDF>
```

Figure 4.15: The semantic annotation, generated automatically to the input annotation of figure 4.14.

agricultural extension.



Create semantic annotation

URL: http://www.lis.ic.unicamp.br/~sidney/carioca_crop.png

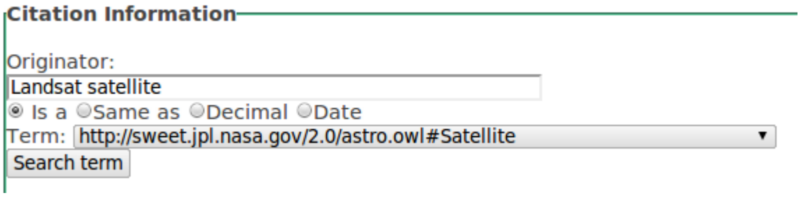
Schema: FGDC + Agricultural Extension

Figure 4.16: Creating a semantic annotation from scratch.

A form with the set of metadata fields that the user has chosen is opened. As said in section 4.2.2, the use of each metadata field is optional, so the user just needs to fill the desired fields. From this point, the user can create semantic annotations in two different manners. In the first, the user can just fill the metadata fields with natural language text and in the end invoke the automatic generation of the corresponding semantic annotation, as just explained. In the second, the user can choose the desired ontology terms for each semantic annotation unit, and specify the kind of relationship between the natural language text and the ontology terms (in the first manner, all relationships are set to "is a").

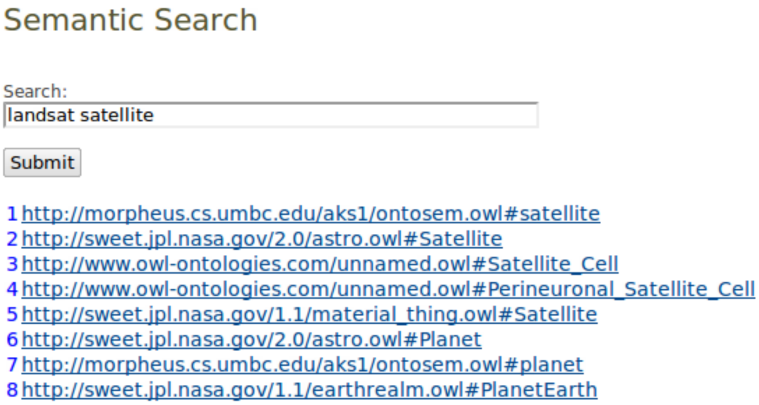
Figure 4.17 shows the creation of the semantic annotation unit corresponding to the *Originator* FGDC metadata field[2]. First, the user types the natural language text "Landsat satellite" at the *Originator* text field, and clicks on the *Search term* button and a list of suggested ontology terms are listed at the *Term* combo box. Figure 4.18 shows all the ranked suggested terms, in a query performed at the page for searching over the repository of tagged ontology terms. The user chooses the *http://sweet.jpl.nasa.gov/2.0/astro.owl#Satellite* ontology term and specifies the type of relationship between the natural language text and the ontology term - in this case, a "is a" relationship. Two extra options are provided, *Decimal* for indicating that the natural language text is a decimal number, and *Date* if the text is a date. It is possible to extend it to provide other options in future implementations - for instance, *String* for indicating that the content filled by the user is a string.

[2] It means the originator of the document, that is, the one that produced/created it.



Originator:
Landsat satellite
● Is a ○ Same as ○ Decimal ○ Date
Term: http://sweet.jpl.nasa.gov/2.0/astro.owl#Satellite ▾

Figure 4.17: Creating a semantic annotation unit.

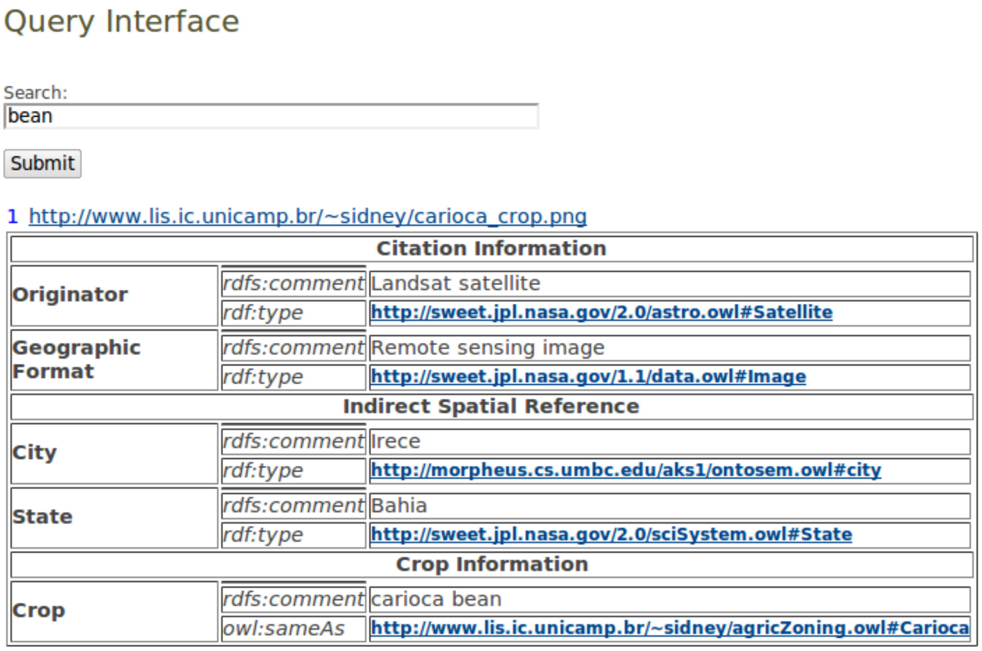


Search:
landsat satellite

1 http://morpheus.cs.umbc.edu/aks1/ontosem.owl#satellite
2 http://sweet.jpl.nasa.gov/2.0/astro.owl#Satellite
3 http://www.owl-ontologies.com/unnamed.owl#Satellite_Cell
4 http://www.owl-ontologies.com/unnamed.owl#Perineuronal_Satellite_Cell
5 http://sweet.jpl.nasa.gov/1.1/material_thing.owl#Satellite
6 http://sweet.jpl.nasa.gov/2.0/astro.owl#Planet
7 http://morpheus.cs.umbc.edu/aks1/ontosem.owl#planet
8 http://sweet.jpl.nasa.gov/1.1/earthrealm.owl#PlanetEarth

Figure 4.18: Suggested ontology terms for “landsat satellite”.

Once all semantic annotation units are created, the user can store them at the repository of semantic annotations for future retrieval. Figure 4.19 shows the interface for querying semantic annotations. The user performs a search - in natural language - for semantic annotations containing some information related to “bean”. The service for querying semantic annotations uses the semantic search service in order to expand the query and to get all ontology terms that are related to it. Thus, the service searches by semantic annotations that contains at least one of the ontology terms returned by the semantic search service. If no terms are returned, the service for querying semantic annotations performs the user’s query over the natural language text of the *rdfs:comment* tags. In the example of figure 4.19, the user clicks on the URL of the remote sensing image annotated in figures 4.16 and 4.17. By clicking on a URL, a form is opened containing the semantic annotation in a more-human readable structure. If the user clicks on the URI of an ontology term of the semantic annotation, a search is performed for semantic annotations whose units contain this term.



Query Interface

Search: bean

1 http://www.lis.ic.unicamp.br/~sidney/carioca_crop.png

Citation Information		
Originator	*rdfs:comment*	Landsat satellite
	rdf:type	**http://sweet.jpl.nasa.gov/2.0/astro.owl#Satellite**
Geographic Format	*rdfs:comment*	Remote sensing image
	rdf:type	**http://sweet.jpl.nasa.gov/1.1/data.owl#Image**
Indirect Spatial Reference		
City	*rdfs:comment*	Irece
	rdf:type	**http://morpheus.cs.umbc.edu/aks1/ontosem.owl#city**
State	*rdfs:comment*	Bahia
	rdf:type	**http://sweet.jpl.nasa.gov/2.0/sciSystem.owl#State**
Crop Information		
Crop	*rdfs:comment*	carioca bean
	owl:sameAs	**http://www.lis.ic.unicamp.br/~sidney/agricZoning.owl#Carioca**

Figure 4.19: Retrieval of a semantic annotation. Clicking on the link returned yields the annotation below

4.4 Tests and Validation

To transform an annotation into a semantic annotation, the service for creating semantic annotations needs to choose appropriate ontology terms for each annotation unit. This section describes the tests performed to validate the implementation of this process.

4.4.1 Choice of Ontology Terms

To validate the service of creation of semantic annotations, we developed a few black-box testing procedures - i.e., designing annotations in which some units had invalid, empty, or erroneous values. Four kinds of annotation units were created to test such situations. First, annotations containing values that could not match any ontology term from the ontology terms repository - and thus, these annotation units could not be mapped to semantic annotation units. Second, we provided inputs with empty annotation units, which therefore could not be mapped to semantic annotation units. Third, we provided inputs where values had typographical errors, which could hamper the choice of appropriate ontology terms. Finally, we created annotation units where the values had a varying number of words, to check whether this would cause variation on the semantic relationship between the value and the chosen ontology term.

The tests performed detected some problems that hamper the process of choosing appropriate ontology terms. Whereas some of these problems were fixed, others were left for future work.

Problem 1: The value in an annotation unit does not correspond to any ontology term

- ***Input provided***: annotation unit $< s, p, v >$ where v does not correspond to a valid ontology term (e.g., $< image.tiff, Crop,$ "$sweetsugar$" $>$);
- ***Expected output***: the service should try to find an ontology term that is semantically associated with the label of the metadata field (in the example, a term associated with *Crop*, for instance, *http://www.lis.ic.unicamp.br/˜sidney/ agricZoning.owl#Crop*);
- ***Output obtained***: the service did not find an ontology term for v and the annotation unit was not be transformed into a semantic annotation unit.

To solve this problem, we have changed the service so that it searches by ontology terms that match the label of the metadata field. Some metadata fields has abbreviated names in the RDF representation - for instance, *fgdc:geoform* (Geographic Format). To implement this solution, the human-readable names of the fields were stored in XML files so that the service can use them to infer ontology terms. Thus, when no ontology terms are returned, the service of transformation of annotations into semantic annotations takes the label of the metadata field in the respective XML file and asks the semantic search service for ontology terms that match it.

Problem 2: Empty annotation unit

- ***Input provided***: annotation unit $< s, p, v >$ where v is empty (e.g., < *image.tiff, Crop,* >);
- ***Expected output***: the service should try to find an ontology term that is semantically associated with the label of the metadata field (in the example, a term associated with *Crop*, for instance, *http://www.lis.ic.unicamp.br/˜sidney/ agricZoning.owl#Crop*);
- ***Output obtained***: in the automatic process of transformation of an annotation into a semantic annotation, the service did not find an ontology term for v - since it was empty - and the annotation unit could not be transformed into a semantic annotation unit.

In order to solve this problem, we adopted the same solution used in problem 1. Thus, if v is empty, the service uses the label of the metadata field to infer an ontology term.

Problem 3: Annotation unit value with typographical error

- ***Input provided***: annotation unit $< s, p, v >$ where v contains typographical errors (e.g., < *image.tiff, fgdc:issue, corp production* >, where v should be "crops production");
- ***Expected output***: the service should try to find an ontology term close to the annotation unit value with typographical errors (in the example, *http://sweet.jpl.nasa.gov/ 1.1/human_activities.owl#CropsProduction*);
- ***Output obtained***: the service tries to find an ontology term that matches the annotation unit value (e.g., *http://morpheus.cs.umbc.edu/aks1/ ontosem.owl #film-corporation*). If it does not find an ontology term, the label of the metadata field is used to infer an ontology term.

We decided to not implement a solution for this, considering that the service cannot infer whether the annotation unit value has typographical errors. For instance, the user may erroneously type a word but this typed word could be an orthographically correct word with different semantics (e.g., “crop” and “corp”). To solve this problem, it would be necessary to implement a spell checker and ask the user about the semantics of the annotation unit value.

Problem 4: Varying number of words in the annotation unit value

- ***Input provided***: annotation units $< s, p, v >$, varying the number of words to compose v;
- ***Expected output***: the service should try to find an ontology term that is semantically associated to the full text from v);
- ***Output obtained***: the service returned an ontology term that is associated to part of the text from v.

We have decided to not implement a solution for this problem, since it needed very specific treatment. Parts of text may design various entities. For instance, the phrase “mulatinho bean produced at Irecê city, Bahia, in the period of october to november” has entities such as “mulatinho bean”, “Irecê”, “Bahia”, and words needed to compose the information contained within the phrase. For the service of semantic search, the greater the number of entities v, the greater the semantic variety of the ontology terms that it will choose. Thus, if the number of words is big, it will be harder to satisfy the user, because each semantic annotation unit has just one ontology term and the service must return just one term that is semantically associated to just one the the entities of the natural language text.

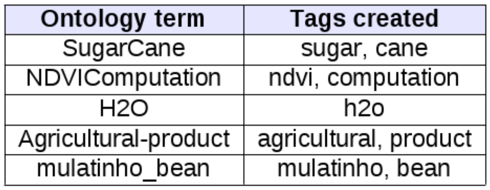
4.4.2 Indexing of Ontology Terms

When an ontology term is extracted, the service of extraction of ontology terms creates tags for indexing it. For this, the service extracts from the URI of the term its name and uses it as a tag. For instance, given the URI of the term *http://www.lis.ic.unicamp.br/˜sidney/agricZoning.owl#Carioca*, the word extracted is “Carioca”. When a term has a composite name, for instance, *http://sweet.jpl.nasa.gov/1.1/human_activities.owl #AgriculturalProduct*, the service extracts the words “Agricultural” and “Product”.

The biggest problem in this process is that there is no accepted pattern for naming ontology terms. In most ontologies, the terms are named following a pattern where the first character of each word must be in uppercase. In this pattern, the term “carioca” is mapped to “Carioca”, “agricultural product” to “AgriculturalProduct”, and “ndvi computation” to “NDVIComputation”. Thus, composite names are identified by the appearance of uppercase characters and initials are written with uppercase characters. However, some ontologies either are designed following other patterns or do not follow any pattern.

This problem was detected during the test phase of the service of extraction of ontology terms. To partially solve this problem, we performed an experiment where several queries for ontologies were performed at Swoogle [71], a Web site for searching ontologies on the Web. The ontologies covered domains like biology, chemistry, geography, agricultural sciences, among others. By performing several queries, and varying the domain of the requested ontologies, we aimed to enhance the chances of finding ontologies with different naming patterns. In this experiment, it was possible to detect two additional patterns for naming ontology terms. In the first pattern, terms with composite names have words separated by '-', whereas in the second pattern words are separated by '_'.

Given these reasons, the service of extraction of ontology terms handles just these three patterns of ontology terms terminology, since they cover the great majority of ontologies available in the Web. Figure 4.20 shows some examples of extraction of tags from ontology terms that were named using the three patterns handled by the service.



Ontology term	Tags created
SugarCane	sugar, cane
NDVIComputation	ndvi, computation
H2O	h2o
Agricultural-product	agricultural, product
mulatinho_bean	mulatinho, bean

Figure 4.20: Examples of extraction of tags.

4.5 Conclusions

This chapter presented the implementation aspects of the architecture of this dissertation. It presented the technologies used in the implementation of the services and the repositories. It was also presented examples of use of the automatic and the user-aided processes of creation of semantic annotations. Finally, it presented the tests performed to identify and solve problems during the processes of management of ontology terms. The next chapter presents the conclusions about this dissertation and future work.

Capítulo 5

Conclusões e Trabalhos Futuros

A heterogeneidade e a distribuição geográfica da informação geo-espacial disponível na *Web* são fatores que dificultam a sua recuperação. Metadados têm sido adotados para diminuir o problema, mas o preenchimento de campos de metadados com texto em linguagem natural e o uso de palavras-chave podem restringir o resultado de consultas e até mesmo levar a ambiguidades. Esta dissertação ataca estes problemas, propondo uma arquitetura para a transformação de anotações em anotações semânticas. Tais anotações semânticas visam melhorar a semântica na descriąão de documentos de conteúdo digital disponíveis na *Web*, associando a estes conceitos não ambíguos que explicam de uma forma mais abrangente os assuntos aos quais se referem.

5.1 Contribuições

Esta dissertação parte do pressuposto que pesquisadores fazem anotações sobre documentos que eles criam ou encontram na *Web*, afim de auxiliar o reuso de tais documentos. Afim de entender tal processo, esta dissertação realizou um levantamento sobre diferentes naturezas de anotação. Isto auxiliou a entender este processo realizado por pesquisadores que necessitam gerar meta-informação a respeito dos artefatos digitais que utilizam.

A arquitetura proposta na dissertação oferece meios para que o pesquisador possa criar anotações sobre tais artefatos, utilizando metadados padronizados, e transformar tais anotações em anotações semânticas. Desta forma, o pesquisador adiciona às suas anotações conceitos da *Web* Semântica, os quais visam fazer com que computadores possam entender conhecimento humano e assim oferecer uma melhor busca pela informação disponível na *Web*.

Para que a arquitetura pudesse oferecer semântica às anotações, foi necessário criar meios para que pesquisadores pudessem incluir suas ontologias preferidas e, com estas já inclusas, pudessem então escolher termos semanticamente relacionados às suas unidades de anotação. Para tanto, foi criado um serviço que permite que o pesquisador possa incluir facilmente ontologias. Fornecida a URL da ontologia, o serviço processa seus termos e lhes adiciona *tags* para facilitar sua busca. Os termos e *tags* são armazenados em um repositório, o qual pode ser utilizado por outras aplicações que necessitem adicionar semântica a dados, utilizando termos de ontologias.

Criadas as anotações semânticas, é necessário persistí-las. Isto acarretou um estudo sobre diferentes mecanismos de armazenamento, onde foram estudados três tipos de soluções: bancos de dados XML nativos, bancos de dados relacionais e bancos de dados RDF. O estudo concluiu que os bancos de dados RDF são a melhor solução, entre as estudadas, para armazenar anotações semânticas. A razão é que bancos de dados RDF oferecem um conjunto de operações pré-definidas para armazenamento, serialização e consulta de dados em RDF - *framework* utilizado nesta dissertação para representar anotações semânticas.

Desta forma, as contribuições desta dissertação são sumarizadas como segue:

- Estudo sobre diferentes naturezas de anotação. Neste estudo, foram levantadas diversas ferramentas para produzir anotações em vários formatos (e.g., texto, áudio, *sketches*, ontologias) sobre conteúdos variados (livros, imagens em 2D e 3D, páginas *Web*). Tal estudo foi necessário para entender as formas em que cientistas anotam seus artefatos para reuso e compartilhamento;

- Especifica
  ccão e implementação de algoritmos para gerenciar termos de ontologias. Tais algoritmos formaram uma base sólida para a processo de criação de anotações semânticas, oferecendo a inclusão de ontologias e busca por termos apropriados;

- Especificação e implementação de uma arquitetura baseada em serviços para gerar e gerenciar anotações semânticas. A arquitetura permite transformar anotações em anotações semânticas utilizando metadados padronizados, de forma automática ou semi-automática. Além disso, permite busca e criação de anotações semânticas;

- A arquitetura também possui um serviço para a extração e indexação de termos de ontologias e um serviço de busca por estes termos;

- Criação de anotações semânticas considerando o domínio agrícola, oferecendo metadados para identificar assuntos relacionados a plantações e questões de produtividade, solo e clima;

- Análise comparativa de mecanismos para armazenamento de anotações semânticas, visando soluções que melhor atendessem a padrões da *Web* Semântica, como RDF e SPARQL.

Comparando tais contribuições com a tabela comparativa do capítulo 2, reproduzida aqui para facilitar a leitura, cabem as seguintes observações adicionais. A tabela 5.1 retrata na última linha, as características do trabalho desta dissertação. A ferramenta implementada é capaz de anotar semanticamente documentos multimídia, onde as anotações criadas são representadas em RDF. Ela utiliza tecnologias da Web Semântica como RDF e ontologias em OWL, além de utilizar um dicionário de sinônimos. As anotações criadas são armazenadas em um banco de dados RDF. Por fim, as anotações semânticas podem ser criadas tanto semi-automaticamente, utilizando teclado e mouse, quanto automaticamente, por meio de um processo que transforma anotações em anotações semânticas.

5.2 Extensões

O estudo realizado neste trabalho permite, dentre outras, as seguintes extensões:

- **Implementação de uma interface gráfica para escolha de termos de ontologias**: no processo de criação semi-automática de anotações semânticas, o usuário digita um conjunto de palavras em linguagem natural e então uma lista de termos de ontologias associados a tais palavras é retornado. Apesar da facilidade deste processo, o usuário teria uma maior facilidade para escolher um termo apropriado se fosse oferecido a ele uma interface que exibisse as ontologias com os termos relacionados às palavras - em forma de grafo. Desta forma, o usuário poderia "navegar" pelas ontologias, entender a hierarquia dos termos das ontologias e escolher um termo; assim, a chance do usuário escolher o termo mais apropriado para a unidade de anotação semântica seria consideravelmente maior;

- **Transformação em lote de anotações em anotações semânticas**: atualmente, o serviço de transformação de anotações em anotações semânticas transforma apenas uma anotação por vez. Porém, o serviço poderia ser modificado para aceitar como entrada adicional e opcional o caminho de um diretório com um conjunto de anotações a serem transformadas em anotações semânticas, tornando assim o serviço mais prático;

- **Uso de axiomas nas unidades de anotação semântica**: nas unidades de anotação semântica, *object* é um termo (classe) de ontologia. Porém, ontologias também possuem axiomas, que são relacionamentos não pré-definidos entre classes. Por exemplo, a classe *Irecê* pode possuir um relacionamento *pertenceAEstado* com a classe *Bahia*. O grande problema de usar tais axiomas é que eles são particulares a cada ontologia, o que exige um processamento mais complexo da ontologia afim de identificar cada axioma e a semântica atribuída a este;

- **Uso de referências a outros documentos nas unidades de anotação semântica**: na tupla $< s, p, o >$, ao invés de uma referência a uma ontologia, o poderia ser a URL de outro documento que já foi semanticamente anotado. Isto criaria uma rede entre anotações semânticas, relacionando e identificando documentos que possuem assuntos em comum. Para isto, seria necessário ampliar o serviço de busca semântica para realizar buscas sobre o repositório de anotações semânticas;

- **Limitação no nível de generalização na indexação de termos de ontologias**: na etapa de extração e indexação de termos de ontologias, cada termo recebe em seu conjunto de *tags* o nome de suas super-classes. Apesar destes nomes serem semanticamente relacionados ao termo, isto pode criar um número excessivo de *tags* quando a ontologia é grande e o termo possui muitas super-classes (ou seja, o caminho entre o termo - a folha - e a sua última super-classe - a raiz - é muito grande). Desta forma, o serviço de indexação de termos poderia receber como parâmetro o número máximo de super-classes a serem utilizadas (ou seja, nível máximo a se subir no caminho entre a folha e a raiz no grafo);

- **Uso controlado de especialização na indexação de termos de ontologias**: se o pesquisador inserir uma ontologia sobre plantações e culturas e realizar uma consulta sobre, por exemplo, “feijâo” na busca semântica, a consulta retornará entre os termos resultantes o termo *Cultura*, pois feijâo é um tipo de cultura. Porém, na consulta, a recíproca não é verdadeira, pois a etapa de indexação considera apenas a generalização entre classes de ontologias - ou seja, *Cultura* não retornará “feijão”. Uma idéia alternativa seria utilizar também a especialização de maneira controlada, utilizando um nível máximo de especialização onde a semântica entre o termo e suas sub-classes pudesse ser mantida, no caso de ontologias grandes;

- **Maior controle na escolha de sinônimos na fase de indexação de termos de ontologias**: cada termo, na etapa de indexação, também recebe como *tags* os seus sinônimos. Porém, no caso de palavras homônimas, pode ocorrer que um ou mais sinônimos não sejam aplicáveis à palavra. Assim, poder-se-ia implementar um maior controle na atribuição de sinônimos, levando em consideração as super-classes do termo na ontologia e desta forma escolhendo apenas os sinônimos semanticamente associados;

- **Implantação do serviço de extração e indexação de termos de ontologias no Aondê [14]**: Como dito na seção 4.2.1, o Aondê apenas oferece busca a ontologias e não termos. Desta forma, o serviço de extração e indexação de termos de ontologias assim como o serviço de busca pelos termos poderiam ser inclusos no Aondê, afim de incluir a este mais esta funcionalidade. Além disso, o SAM poderia se beneficiar do serviço de busca por ontologias do Aondê, em um cenário onde o usuário fizesse uma consulta por um termo de ontologia que ainda não existe no repositório de termos. Assim, o Aondê poderia retornar uma ou mais ontologias, cujos termos poderiam ser extraídos pelo serviço de extração de termos do SAM;

- **Controle sobre diferentes anotações semânticas sobre um mesmo documento**: quando uma anotação semântica é armazenada, a URL do documento ao qual ela se refere é utilizada como chave primária. Porém, se já existir uma anotação semântica sobre um determinado documento e outro usuário quiser criar outra anotação semântica sobre ele, a arquitetura não permite a inclusão desta anotação, pois ocorreria um problema de integridade. Assim, uma possível extensão seria a implementação de um mecanismo para permitir anotações semânticas sobre um mesmo documento, evitando problemas de integridade no repositório;

- **Implementação de um repositório semântico para as anotações semânticas**: a arquitetura implementada nesta dissertação oferece apenas busca por anotações semânticas. Porém, um repositório semântico poderia utilizar as anotações semânticas para oferecer fácil integração entre dados e análises poderosas, utilizando conceitos da *Web* Semântica [58]. Uma solução mais prática para isto seria utilizar o *framework* Sesame [7] - que foi utilizado como repositório de anotações semânticas nesta dissertação - associado ao OWLIN [57], uma camada de inferência que funciona sobre o Sesame.

Tool	**Annotated Data**	**Format**	**SW Technologies**	**Storage**	**Input Method**
bibPhone	books	voice	-	database	microphone from the gadget
Boom Chameleon	3D images	sounds, drawings and sketches	-	different files for each kind of annotation	touch screen device
Digital Graffite	posts at the Plasma Posters Network portal	sketches and voice	-	portal's database	PDA (using stylus pen), keyboard, and mouse
CommonSpace	text	text and voice	-	local file	keyboard and microphone
XLibris	text	highlight and sketches	-	local file	pen
Space Pen	3D images	text and sketches	-	file	keyboard and mouse
eTrace	2D and 3D documents	sketches	-	separate file	pen
Sierra	2D images	text and marks	-	database	keyboard and mouse
BOEMIE	text, Web pages	ontologies	OWL ontologies	OWL or XML file	keyboard and mouse
DocSS	text documents	metadata fields	RDF	Sesame RDF database	keyboard and mouse
Repp *et all* [63]	video	ontologies	OWL ontologies	OWL file	keyboard and mouse
Warner and Chun [75]	multimedia documents	tags	Ontologies and taxonomies	Tags repositories	keyboard and mouse
Annotea	Web pages	comments and notes	RDF	RDF code at a server	keyboard and mouse
SAM	**multimedia documents**	**RDF annotation**	**RDF, OWL ontologies, thesaurus**	**Sesame RDF database**	**automatic / keyboard and mouse**

Tabela 5.1: Comparação entre as ferramentas estudadas e o trabalho proposto pela dissertação

Referências Bibliográficas

[1] Sun Microsystems. Java Servlet Technology. Available in http://java.sun.com/products/servlet/. Accessed in November 24th, 2009.

[2] 3M. Post-it Digital Products. Available in http://www.3m.com/us/office/postit/digital/. Accessed in December 10th, 2009.

[3] M. A. Angrosh and S. R. Urs. Development of Indian Agricultural Research Ontology: Semantic Rich Relations Based Information Retrieval System for Vidyanidhi Digital Library. In *Lecture Notes in Computer Science*, pages 400–409. Springer Berlin / Heidelberg, 2007.

[4] Annozilla. Annozilla (Annotea on Mozilla). Available in http://annozilla.mozdev.org/. Accessed in December 15th, 2009.

[5] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. In *Scientific American Magazine*, pages 34–43, May 2001.

[6] A. D. Blaser. A Study of People's Sketching Habits in GIS. *Spatial Cognition and Computation*, 2(4):393–419, 2001.

[7] J. Broekstra, A. Kampman, and F. van Harmelen. Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. pages 54–68. Springer Berlin / Heidelberg, 2002.

[8] H. Brugman, V. Malaisé, L. Gazendam, and G. Schreiber. The Documentalist Support System: a Web-Services based Tool for Semantic Annotation and Browsing. In *Semantic Web Challenge 2008*, 2008.

[9] P. Cano. Automatic Sound Annotation. In *IEEE workshop on Machine Learning for Signal Processing*, pages 391–400, 2004.

[10] Carnegie Mellon University. PREP Editor. Available in http://eserver.org/software/prep/. Accessed in August 08th, 2008.

[11] S. Carter, E. Churchill, L. Denoue, J. Helfman, and L. Nelson. Digital Graffiti: Public Annotation of Multimedia Content. In *CHI '04: CHI '04 extended abstracts on Human factors in computing systems*, pages 1207–1210, New York, NY, USA, 2004. ACM.

[12] A. Chandler and D. Foley. Mapping and Converting Essential Federal Geographic Data Committee (FGDC) Metadata into MARC21 and Dublin Core: Towards an Alternative to the FGDC Clearinghouse. In *D-Lib Magazine*, volume 6, 2000.

[13] W. W. Cohen, P. Ravikumar, and S. E. Fienberg. A Comparison of String Distance Metrics for Name-Matching Tasks. In *IIWeb*, pages 73–78, 2003.

[14] J. Daltio and C. B. Medeiros. Aondê: An Ontology Web Service for Interoperability across Biodiversity Applications. *Information Systems*, 33(7-8):724–753, 2008.

[15] S. R. de Sousa. A Semantic Approach to Describe Geospatial Resources. In *3rd International Workshop on Semantic and Conceptual Issues in GIS (SeCoGIS 2009)*, volume LNCS 5833, pages 327–336, November 2009.

[16] S. R. de Sousa and C. B. Medeiros. Management of Semantic Annotations of Data on Web for Agricultural Applications. In *VIII WTDBD - Workshop de Teses e Dissertações em Bancos de Dados*, October 2009.

[17] S. Dill, N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, S. Rajagopalan, A. Tomkins, J. A. Tomlin, , and J. Y. Zien. SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation. In *WWW 2003*, pages 178–186. ACM Press, 2003.

[18] E. Y. Do and M. D. Gross. As if You Were Here- Intelligent Annotation in Space: 3D Sketching as an Interface to Knowledge-Based Design Systems. American Association for Artificial Intelligence, 2004.

[19] eXist. Open Source Native XML Database. Available in http://www.exist-db.org/. Accessed in October 12th, 2009.

[20] Federal Geographic Data Committee. Alphabetical List of Compound Elements and Data Elements - Federal Geographic Data Committee. Available in http://www.fgdc.gov/metadata/csdgm/list.html. Accessed in October 29th, 2009.

[21] C. Fellbaum. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press, May 1998.

[22] D. Fensel, F. V. Harmelen, I. Horrocks, D. L. McGuinness, and P. F. Patel-Schneider. OIL: An Ontology Infrastructure for the Semantic Web. *Inteligente Systems, IEEE*, 16(2):38–45, Mar-Apr 2001.

[23] F. Fonseca and A. Rodriguez. From Geo-Pragmatics to Derivation Ontologies: new Directions for the GeoSpatial Semantic Web. *Transactions in GIS*, 11(3):313–316, 2007.

[24] Food and Agriculture Organization of the United Nations. GeoNetwork. Available in http://www.fao.org/geonetwork/srv/en/main.home. Accessed in December 19th, 2009.

[25] P. Fragkou, G. Petasis, A. Theodorakos, V. Karkaletsis, and C. Spyropoulos. BOEMIE ontology-based text annotation tool. In *Proceedings of the Sixth International Language Resources and Evaluation LREC08*, pages 1273–1279, 2008.

[26] R. B. Freitas and R. da S Torres. OntoSAIA: Um Ambiente Baseado em Ontologias para Recuperação e Anotação Semi-Automática de Imagens. In *Primeiro Workshop de Bibliotecas Digitais, Simpósio Brasileiro de Banco de Dados*, pages 60–79, Uberlandia, MG, Brazil, October 2005.

[27] FxPal. Plasma Poster Network. Available in http://www.miracosta.edu/home/gfloren/OT-CS.htm. Accessed in August 08, 2008.

[28] D. Gorgan, T. Stefanut, and B. Gavrea. Pen Based Graphical Annotation in Medical Education. In *CBMS '07: Proceedings of the Twentieth IEEE International Symposium on Computer-Based Medical Systems*, pages 681–686, Washington, DC, USA, 2007. IEEE Computer Society.

[29] T. R. Gruber. A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

[30] T. R. Gruber. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. *Internation Journal Human-Computer Studies*, 43:907–928, 1993.

[31] J. Hendler and D. McGuinness. The DARPA Agent Markup Language. *IEEE Inteligent Systems*, 15(6):67–73, 2000.

[32] International Organization for Standardization. ISO 19115:2003. Available in http://www.iso.org/iso/isoc_atalogue/catalogue_tc/catalogue_detail.htm?csnumber=26020. Accessed in December 09, 2009.

[33] M. A. Jaro. Probabilistic Linkage of Large Public Health Data Files. In *Statistics in Medicine*, volume 14, pages 491–498, 1995.

[34] T. Jung, E. Y. Do, and M. D. Gross. From Redliner to Space Pen. In *ACM Intelligent User Interfaces*, pages 95–102. ACM Press, 2002.

[35] J. Kahan and M.-R. Koivunen. Annotea: an open RDF infrastructure for shared Web annotations. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 623–632, New York, NY, USA, 2001. ACM Press.

[36] O. Karschnick, F. Kruse, S. Topker, T. Riegel, M. Eichler, and S. Behrens. The UDK and ISO 19115 Standard. In *Proceedings of the 17th International Conference Informatics for Environmental Protection EnviroInfo*, 2003.

[37] E. Klien. A Rule-Based Strategy for the Semantic Annotation of Geodata. In *Transactions in GIS*, volume 11, pages 437–452, 2007.

[38] E. Klien and M. Lutz. The Role of Spatial Relations in Automating the Semantic Annotation of Geodata. In A. Cohn and D. Mark, editors, *Spatial Information Theory: International Conference, COSIT 2005,Ellicottville, NY, USA, September 14-18, 2005*, Lecture Notes in Computer Science (LNCS), pages 133–148. Springer, 2005.

[39] Laboratory of Information Systems - IC - UNICAMP. About eFarms. Available in http://proj.lis.ic.unicamp.br/efarms. Accessed in December 10th, 2009.

[40] M. C. Lange, D. G. Lemay, and J. B. German. A Multi-ontology Framework to Guide Agriculture and Food Towards Diet and Health. In *Journal of the Science of Food and Agriculture*, volume 87, pages 1427–1434, 2007.

[41] M. Lesaffre, K. Tanghe, G. Martens, D. Moelants, M. Leman, B. D. Baets, H. D. Meyer, and J.-P. Martens. The MAMI Query-By-Voice Experiment: Collecting and Annotating Vocal Queries for Music Information Retrieval. In *Proceedings of the 4th International Conference on Music Information Retrieval*, pages 65–71, 2003.

[42] A. C. Liang, M. S. Boris Lauser, J. Keizer, and S. Katz. From AGROVOC to the Agricultural Ontology Service / Concept Server: An OWL model for managing ontologies in the agricultural domain. In *OWL: Experiences and Directions Workshop Series*, 2006.

[43] T. Libourel, I. Mougenot, J. Sallantin, and L. Spéry. Meta-Data and Biological Sequence Annotation. In *META-DATA 99 : 3rd IEEE META-DATA Conference, National Institutes of Health Natcher*, 1999.

[44] C. Ljunggrens. Janno. Available in http://www.claus-ljunggren.dk/janno/web/index.html. Accessed in December 15th, 2009.

[45] A. Lykke-Olesen and J. Nielsen. bibPhone: Adding Sound to the Children's Library. In *IDC '07: Proceedings of the 6th international conference on Interaction design and children*, pages 145–148, New York, NY, USA, 2007. ACM.

[46] C. G. N. Macário. *Specification of a Framework for Semantic Annotation of Geospatial Data on the Web.* PhD thesis, Institute of Computing, UNICAMP, 2008.

[47] C. G. N. Macário, S. R. de Sousa, and C. B. Medeiros. Annotating Geospatial Data based on its Semantics. In *17th ACM SIGSPATIAL Conference*, pages 81–90, Seattle, USA, November 2009. ACM.

[48] C. G. N. Macário and C. B. Medeiros. A Framework for Semantic Annotation of Geospatial Data for Agriculture. *Int. J. Metadata, Semantics and Ontology - Special Issue on "Agricultural Metadata and Semantics"*, 2008. Accepted for publication.

[49] D. Maier and L. Delcambre. Superimposed Information for the Internet. In *Proceedings of the WebDB Workshop*, pages 1–9, Philadelphia, PA, USA, 1999.

[50] M. T. Maliappis. Technological Aspects of Using Agricultural Ontologies. In *Proceedings of the 2nd International Conference on Metadata and Semantics Research (MTSR'07)*, Corfu, Greece, October 2007.

[51] MODIS Department. Sedna XML Database. Available in http://modis.ispras.ru/sedna/. Accessed in October 12th, 2009.

[52] J. Mori, Y. Matsuo, M. Ishizuka, and B. Faltings. Keyword Extraction from the Web for Personal Metadata Annotation. In *ISWC Workshop Notes VIII 115 (W8) - 4th International Workshop on Knowledge Markup and Semantic Annotation (Semannot2004) (in conjunction with 3rd Internationall Semantic Web Conference (ISWC2004))*, pages 51–60, 2004.

[53] S. Murthy, D. Maier, L. Delcambre, and S. Bowers. Putting Integrated Information into Context: Superimposing Conceptual Models with SPARCE. In *Proceedings of the First Asia-Pacific Conference of Conceptual Modeling*, pages 71–80, Denedin, New Zealand, 2004.

[54] U. Murthy, R. S. Torres, and E. A. Fox. SIERRA: A Superimposed Application for Enhanced Image Description and Retrieval. In *Lecture Notes in Computer Science : Research and Advanced Technology for Digital Libraries*, pages 540–543, Berlin / Heidelberg, 2006. Springer.

[55] J. Nogueras-Iso, F. J. Zarazaga-Soria, J. Lacasta, R. Bejar, and P. R. Muro-Medrano. Metadata Standard Interoperability: Application in the Geographic Information Domain. *Computers, environment and urban systems*, 28(6):611–634, 2003.

[56] OnTap. Evaluation of COMMON SPACE. Available in http://www.miracosta.edu/home/gfloren/OT-CS.htm. Accessed in December 10th, 2009.

[57] ontotext Semantic Technology Lab. OWLIM Semantic Repository. Available in http://www.ontotext.com/owlim/. Accessed in December 3rd, 2009.

[58] ontotext Semantic Technology Lab. Semantic Repository. Available in http://www.ontotext.com/inference/semantic_repository.html. Accessed in December 3rd, 2009.

[59] E. Oren, K. Moller, S. Scerri, S. Handschuh, and M. Sintek. What are Semantic Annotations? Technical report, DERI Galway, 2006.

[60] G. Z. Pastorello Jr, J. Daltio, and C. B. Medeiros. Multimedia Semantic Annotation Propagation. In *Proceedings of the 1st IEEE International Workshop on Data Semantics for Multimedia Systems and Applications (DSMSA) – 10th IEEE International Symposium on Multimedia (ISM)*, December 2008.

[61] B. Popov, A. Kiryakov, A. Kirilov, D. Manov, and O. M. Goranov. KIM - Semantic Annotation Platform. In *In Proc. of the 2 nd Intl. Semantic Web Conf. 2003*, pages 834–849. Springer, 2003.

[62] M. N. Price, B. N. Schilit, and G. Golovchinsky. XLibris: the active reading machine. In *CHI' 98: Conference Summary on Human factors in Computing Systems*, pages 22–23, New York, NY, USA, 1998. ACM.

[63] S. Repp, S. Linckels, and C. Meinel. Towards to an Automatic Semantic Annotation for Multimedia Learning Objects. In *Emme '07: Proceedings of the international workshop on Educational multimedia and multimedia education*, pages 19–26, 2007.

[64] RFID Journal. RFID, What is RFID, History of RFID. Available in http://www.rfidjournal.com/article/gettingstarted. Accessed in December 10th, 2009.

[65] Stanford University. The Protégé Ontology Editor and Knowledge Acquisition System. Available in http://protege.stanford.edu/. Accessed in December 10th, 2009.

[66] M. Suwa and B. Tversky. What Architects See in their Sketches: Implications for Design Tools. In *CHI '96: Conference companion on Human factors in computing systems*, pages 191–192, New York, NY, USA, 1996. ACM.

[67] The Apache Software Foundation. Apache Lucene - Overview . Available in http://lucene.apache.org/java/docs/. Accessed in November 24th, 2009.

[68] R. Torres, C. B. Medeiros, M. A. Goncalves, and E. A. Fox. An OAI Compliant Content-based Image Search Component. In *Proceedings of the 4th ACM/IEEE-CS joint conference on digital libraries*, pages 3–17, Tuscon, AZ, USA, 2004. ACM Press.

[69] M. Tsang, G. W. Fitzmaurice, G. Kurtenbach, A. Khan, and B. Buxton. Boom Chameleon: simultaneous capture of 3D viewpoint, voice and gesture annotations on a spatially-aware display. In *UIST '02: Proceedings of the 15th annual ACM symposium on User interface software and technology*, pages 111–120, New York, NY, USA, 2002. ACM.

[70] Universität Konstanz. BaseX: XML Database and XPath/XQuery Full Text Processor. Available in http://www.inf.uni-konstanz.de/dbis/basex/. Accessed in October 12th, 2009.

[71] University of Maryland, Baltimore County. Swoogle Semantic Web Search Engine. Available in http://swoogle.umbc.edu/. Accessed in January 14th, 2010.

[72] M. Uschold and M. Gruninger. Ontologies: Principles, Methods, and Applications. *Knowledge Engineering Review*, 11(2):93–155, 1996.

[73] M. Vargas-Vera, E. Motta, J. Domingue, S. B. Shum, and M. Lanzoni. Knowledge Extraction by Using an Ontology-based Annotation Tool. In *K-CAP 2001 Workshop on Knowledge Markup and Semantic Annotation*, pages 5–12, 2001.

[74] W3 Schools. AJAX Tutorial. Available in http://www.w3schools.com/Ajax/Default.Asp. Accessed in November 24th, 2009.

[75] J. Warner and S. A. Chun. Semantic and pragmatic annotation for government information discovery, sharing and collaboration. In *dg.o '09: Proceedings of the 10th Annual International Conference on Digital Government Research*, pages 199–205, 2009.

[76] Web3D Consortium. VRML97 and Related Specifications. Available in http://www.web3d.org/x3d/specifications/vrml/. Accessed in December 10th, 2009.

[77] S. L. Weibel and T. Koch. Dublin Core Metadata Initiative: Mission, Current Activities, and Future Directions. *D-Lib Magazine*, 6(12), 2000.

[78] K. Wilkinson, C. Sayers, H. Kuno, and D. Reynolds. Efficient RDF Storage and Retrieval in Jena2. In *EXPLOITING HYPERLINKS 349*, pages 35–43, 2003.

[79] J. Wolfe. Annotation Technologies: A Software and Research Review. *Computers and Composition*, 19(4):471–497, December 2002.

[80] World Wide Web Consortium. annoChump Overview. Available in http://www.w3.org/2001/09/chump/. Accessed in December 15th, 2009.

[81] World Wide Web Consortium. Notation3 (N3): A readable RDF syntax. Available in http://www.w3.org/DesignIssues/Notation3. Accessed in October 12th, 2009.

[82] World Wide Web Consortium. RDF/XML Syntax Specification (Revised). Available in http://www.w3.org/TR/rdf-syntax-grammar/. Accessed in October 12th, 2009.

[83] World Wide Web Consortium. RDQL - A Query Language for RDF. Available in http://www.w3.org/Submission/RDQL/. Accessed in October 12th, 2009.

[84] World Wide Web Consortium. Resource Description Framework (RDF). Available in http://www.w3.org/RDF/. Accessed in November 24th, 2009.

[85] World Wide Web Consortium. SPARQL Query Language for RDF. Available in http://www.w3.org/TR/rdf-sparql-query/. Accessed in October 12th, 2009.

[86] World Wide Web Consortium. Web Ontology Language (OWL). Available in http://www.w3.org/2004/OWL/. Accessed in November 24th, 2009.

[87] World Wide Web Consortium. Welcome to Amaya. Available in http://www.w3.org/Amaya/. Accessed in December 15th, 2009.

[88] H. Xie, X. Wu, X. Yu, J. Li, and Y. G. C. G. Yang. Agriculture Emergency Decision System Based on Semantic Web Services. *International Conference on Convergence Information Technology - ICCIT 2007*, pages 503–507, 2007.