Building Yearbooks with RDF

Ernesto Krsulovic Morales, Claudio Gutiérrez
Center for Web Research
Dept. of Computer Science, Universidad de Chile
Blanco Encalada 2120, Santiago, Chile
E-mail: {cgutierr,ekrsulov}@dcc.uchile.cl

Abstract. We present a simple application of semantic integration using the RDF model of metadata, namely, the construction and maintenance of a yearbook. It can be built and used by organizations which already have their information on the Web and require to keep yearbooks to service advanced searching facilities. Unlike traditional approaches, ours ensures wide interoperability, extensibility and historical recording by using RDF and a decentralized approach.

Keywords: Semantic Integration, RDF, Semantic Web, Yearbook

1 Introduction

Yearbooks A yearbook is a periodic document issuing information (reports, statistics, etc.) about a particular subject, and is becoming a common practice to publish them in digital format on the Web. Examples are annual reports published by companies refering to departments, projects, financial information (e.g. search `site:cl memoria anual` in Google for examples from Chile), or association yearbooks, which contain information about the structure of each member organization associated along with activities it develops (e.g. search `''association yearbook''`, in Google for representative examples.) From a semantic point of view, a classic yearbook is directed to a restricted set of users and usually the process of consulting it is mainly a human-oriented task.

Building yearbooks is a typical task in many organizations which, although straightforward at first sight, becomes complicated when the organization has strong hierarchies or several horizontal components. The classic approach to building these information centers is to delegate the task to one of the associated members, which becomes in charge of designing or updating the schema of the data, discussing it with other members, collecting the information among the different components of the organization, and populating the database. *Designing* the schema for a yearbook is a typical problem in the area of integration of information. The weakness of these traditional methodologies, though, is that they are not designed to achieve semantic persistence and extensibility, e.g. make the yearbook usable for people and applications we do no have in mind today [1]. *Collecting* the information to build a yearbook is a task not easy to automatize: each component must, on the one hand, transform the information kept to the common schema chosen for the yearbook, and on the other, create the information needed which is not kept explicitly. If the organization is highly hierarchical, the above task is usually simplified because the organization of the information is predefined for all the organization from the top.

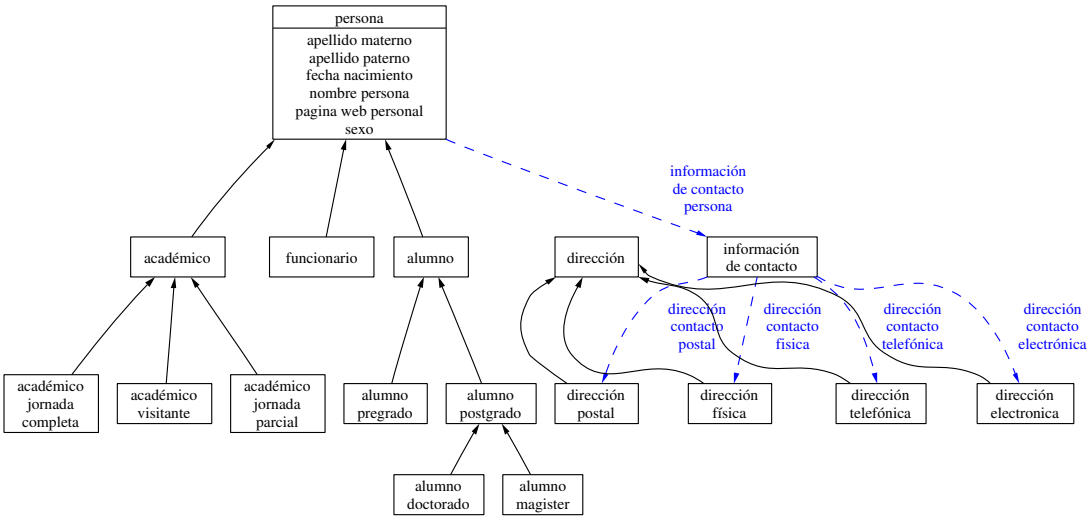

Figure 1: Fragment of the ontology corresponding to people in a university Department.

One approach to overcome the above difficulties is to use semantic integration techniques to design the yearbook, very much like in organizational memory [2], and semi-automatize the process of collecting the information via techniques developed in the framework of the Semantic Web, that is to publish periodically metadata according to an established ontology. Let us describe these frameworks.

Semantic Integration and Decentralization We claim that the construction of a yearbook of organizations on the Web is better implemented by using the framework of semantic integration [3], [4], [5] and a distributed approach [6].

To get its full potential, the information on the Web needs to be interoperable not only at a syntactic or structural level: the semantic layer is an essential one. Although this idea is at the core of the future Semantic Web [7], it is not easy to implement today. What are the tools we have at hand? It is important to note that XML was designed to provide interoperability at structure and document level. XML helps to give semantics to documents via tags with "meaning"; although this is a great advance over HTML, this "meaning" is still very much directed to humans (i.e. difficult or impossible to understand by software agents) and expresses mainly the structure of documents. For a particular domain, XML can be a good choice to express meaning and define common vocabularies trough XML Schema [8]. But when it comes to interoperability of applications across several independent domains, the limitations of XML becomes apparent [3], [9]. The solution proposed by the W3C is a framework for semantic interoperability, namely RDF, described below.

There are serveral schemas of cataloguing on the Web, probably the most popular are indexes like *Yahoo*. Problems with such indexing schemas are the tradeoff between the number of sites indexed and the precision and type of the description (at document level), and the fact that creation and maintenance of metadata is kept centralized. Although indexes such as DMOZ [10] have aleviated cataloguing labor by using voluntary editors by categories, still persist the problems associated to centralized maitenance and storing, like the size of the data, reliability, general descriptions at document level.

A solution to the problem of centralization is to provide the necessary infrastructure to

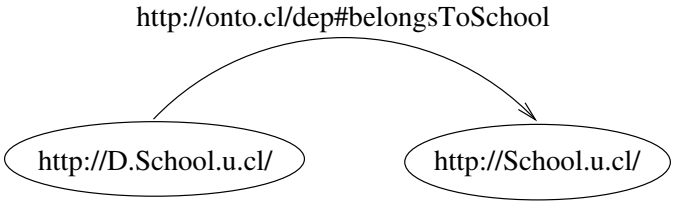

Figure 2: Example of an RDF graph

allow each site to keep its own metadata. At metalanguage level the solution recommended by the W3C is RDF [11]. The solution to the problem of quality of descriptions is the creation of shared vocabularies by topics or domains and a schema to exchange those vocabularies.

The RDF framework *RDF (Resource Description Framework)* is the metadata model and framework proposed by the W3C, and is rapidly gaining popularity as a de facto standard [11]. Using RDF instead of plain XML (not to mention HTML) offers several advantages: (1) a better model from a semantic point of view; (2) full semantic extensibility; (3) facilities to handle distributed data. All these features are very desirable when building a distributed and extensible yearbook. On the Web, data are documents, multimedia objects and links; these data are all called resources. RDF metadata is simply a set of statements about resources using predicates from shared vocabularies. The RDF model strongly supports extensibility. A relational or XML database has a very static schema and it is very difficult to make any significant changes without impacting existing code. Compared to XML, the RDF model is dynamic and it is easy to add new information to an existing description. Moreover, when keeping metadata with RDF it is possible to scale solutions by integration of previously disjoint organizations.

RDF offers a better semantic model than existing alternatives due to the simplicity of its model (a labeled graph where order is not relevant) and its flexibility to describe data (a simple subjet-predicate-object model) [8]. In RDF, resources are identified by a Uniform Resource Identifier (URI), an identification schema which generalizes URLs. In an RDF statement predicates as well as subjects are URIs and the object is a URI or a literal. So for example, it is possible to register the statement "Department D belongs to School S" with the following triple:

```
<http://onto.cl/dep#belongsToSchool>
<http://D.School.u.cl/>
<http://School.u.cl/>
```

where the first is the predicate's URI indicating to what School belongs the given department, the second identifies the department, and the third URI identifies the School. The graph corresponding to this statement is shown in Figure 2.

To describe information on the Web with RDF, we need a common vocabulary, i.e. an *ontology*. One such example is Dublin Core, a minimal vocabulary to describe title, author, copyrights, etc. of a document [12]. For our purposes we need more specific ontologies. RDF allows the specification of such ontologies through RDF Schema (RDFS), which is modelled by classes, sub-classes, hierarchies, relations and properties of classes [13]. For example, in our prototype Yearbook which models university Departments, we need to model people

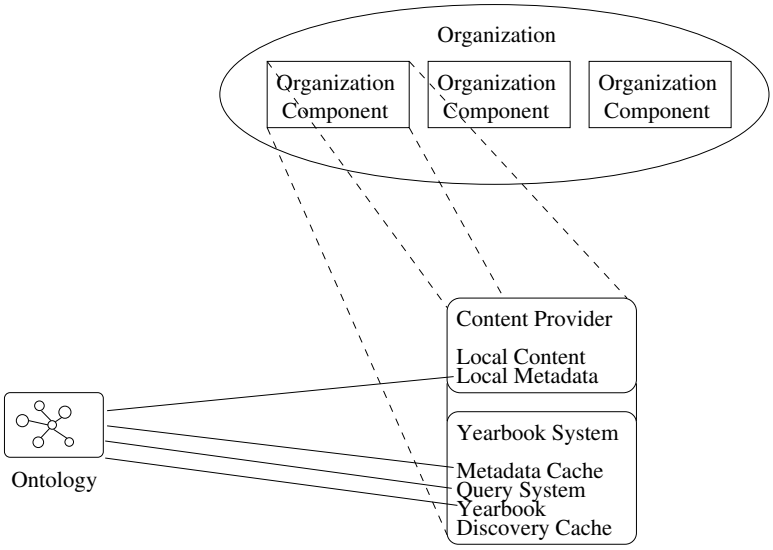

Figure 3: Yearbook system and sites of the organization components.

who belong to a Department and their contact addresses, obtaining a hierarchy of classes, attributes of the class *persona* and the relations among classes as shown in Figure 1.

Handling distributed data with RDF is simple because, as described above, an RDF specification is a collection of atomic sentences over a shared common vocabulary built by different parties. Although storing atomic sentences over diverse vocabularies currently can be a drawback when the size of the data is huge, this is an issue being extensively investigated and resembles the problem of storing efficiently relational data in the early seventies or XML data at the end of the nineties.

Yearbooks versus Catalogs: t-books To finalize the introductory discussion, let us make some remarks about the difference between a yearbook and a catalog in our context and some tips learned about temporality of metadata.

A catalog is a list of items, usually ordered, whose goal is describing data in a succint way, e.g. exhibition catalogs, library catalogs, shopping catalogs. The main implicit characteristic of a classical catalog is the hidden assumption about the invariability (atemporality) of the data it describes. (This is more obvious in the shopping catalogs, where price, season, etc. are key parameters). On the Web, catalogs turned exactly into its opposite: although they still do not have a temporal parameter, its information –in the form of links to web sites– is highly variable. In fact, any attempt to describe data on the Web faces the problem of (extremely and unpredictable) variability of data. For many temporal applications this is not important: the catalog of *amazon.com* changes daily, and probably this is a feature more than a problem. But for other kind of information this is a crucial point. Consider for example a catalog of graduate students or financial statistics.

Yearbooks can be considered essentially catalogs with a fixed time-stamp (is better to call them *t-books*) and secondly, less important for our discussion, that contain aggregate data (statistics). The periodicity of the yearbook usually indicates an actual change of data: new year, new semester, new month, new balance, etc. But several applications need a different timing: bi- or tri-annual for university courses, montly for accounting, daily for newspapers.

We think the concept of yearbooks is a good "discrete approximation" to an online catalog on the Web, useful for several applications, and has the additional advantage that, caching mediating, can be used as historical record, hence providing temporal information usually lost in current "on-line catalogs". From the point of view of the Semantic Web, it has additional advantages: (1) allows historical recording, hence providing temporal information; (2) simplifies marking (with metadata) by restricting it to a smaller group of people, hence more specialized. From a different point of view, a t-book is a compromise between a centralized system like DMOZ and completely distributed system like Edutella [14].

In what follows, we will use "yearbook" to mean a t-book for some $t > 0$.

2 A decentralized Yearbook System

2.1 Arquitecture

Each component of the organization can offer two services: *Content Provider*, that is, provides the information contained in its Web site, and *Yearbook System*, an optional service for the components, although it must be offered at least by one of them (see Figure 3).

The modules of a Content Provider are:

- *Local Content.* Typically static or dynamic Web pages, documents and multimedia content. This information is already in the Web site of each component.
- *Local Metadata.* Markup of local contents in the form of RDF triples, using ontologies defined by the organization. Typically these metadata will not be in the Web site of the component, and must be generated according to the periodicity defined by the designers of the Yearbook.

The Yearbook System gives the following services to the system:

- *Metadata Cache.* Stores copies of the metadata of producers. The metadata is periodically updated according to the system schedule, and keeps historical records of previous versions.
- *Query System.* Provides query functionality to the global metadata of the system. This module is one of the building blocks of the whole system.
- *Yearbook.* Allows "navigation" among metadata and does the processing of predefined queries defined by a hierarchycal structure or a fixed taxonomy. Additionally it provides a searching-by-pattern facility at different levels.
- *Discovery Cache.* Keeps a list of known producers. This module allows new yearbook services to get into the system through one of the components already connected to the network.

2.2 A case of study

We will present a prototype yearbook for the academic Departments of Chilean universities. This domain is atractive because the components (university departments) have great interest in such a system, the organization (Council of Universities, Ministry of Education) have

shown interest in incorporating new tecnologies to their system. Additionally, this is a case where the organization is highly horizontal, with weak leadership, but for students, educators and researchers this group of components form a very natural unit. We started with the particular case of Computer Science Departments because they are a manageable number in Chile, all of them have Web sites, and obviously can participate more actively in the project which involves technologies familiar to them. It is interesting to remark that today we do not have in Chile any similar system, centralized or not, which performs the functionalities we have described for a yearbook. The need for such a system is out of discussion.[1]

Ontology The process of design (or reuse and adaptation, if it is available) of an ontology is among the most important aspects of the project and *is the semantic glue among the modules of the proposed decentralized Yearbook*. We used *Protégé* [15] and followed classical methodologies [16]: building the vocabulary by filtering it from the Web pages of the Chilean Departments of Computer Science, organizing it in classes, determining attributes, relationships and properties. Then we iterated this process through each of the Web pages of the CS Departments. The design is still a preliminar one, and we believe it can be the basis for the ontology of chilean university Departments.

It is worth mentioning that the Academia has been used before as a test bed for markup projects, and university ontologies exist. Unfortunately for us, their concepts, vocabulary and emphasis are far from those used in departments in Chile. Considering this, the best solution we found was to build a local ontology and then specify, at the right level, the conceptual translations to other university ontologies. Below is a brief description of the Ontology. For details see [17].

- Vocabulary for a university Department: 5 abstract classes, 16 classes and 54 attributes to mark admission, teaching and research information, and to describe people belonging to it: students, academic and support staff.
- Vocabulary to describe contact information: 1 abstract class, 5 classes, 17 attributes. It allows to basically specify personal and institutional addresses in four different formats: electronic, physical, telephonic and regular mail.
- Taxonomy of research areas: 28 abstract classes, enough to catalog research areas of people, institutions and research centers existing in Chile (2002).

Local Content Most of the information required by the Yearbook is already in the Web site of each Department in HTML format.

Local Metadata We used Protégé as markup tool. Protégé provides forms to create instances given an ontology and has a facility to export metadata in RDF format. We remark that this tool, as several others available today, requires some training to get all its functionality, particularly the markup process. In the case of Yearbooks, this is not a big problem because usually organizations which build Yearbooks already have dedicated personnel, hence there is no extra investement or effort to orient them to learn this new process.

[1] Today a person needs to navigate manually through a highly heterogeneous set of Web sites to find elementary aggregate information about these departments.

Local metadata is stored in one or few central RDF files that contain all instances of classes for all pages of each Web site of each Department.

Metadata Cache In our current implementation we do not keep cache of metadata. Our experience shows that this is a necessity to scale the system and give it a minimum of efficiency. Our plan is to store metadata in a relational database using the feature of *SQL backend support* of Inklink [18].

Query system The query system implemented provides the functionality of SquishQL [19], a query language over RDF allowing queries very much like SQL. We chose this alternative among several query languages available because it provides a base implementation easy to fit our goals and allows queries on several RDF sources at one time by just writing up different URIs in the FROM predicate. See Figure 4.

```
SELECT ?curso, ?departamento, ?univ
FROM
 http://purl.org/net/depmark/puc,
 http://purl.org/net/depmark/uch,
 http://purl.org/net/depmark/umag,
 http://purl.org/net/depmark/udec,
 http://purl.org/net/depmark/utfsm,
 http://purl.org/net/depmark/usach
WHERE
 (depmark::cursos_del_departamento ?docencia ?idcurso)
 (depmark::nombre_curso ?idcurso ?curso)
 (depmark::docencia_departamento ?iddep ?docencia)
 (depmark::nombre ?iddep ?departamento)
 (depmark::universidad_a_que_pertenece ?iddep ?univ)
USING depmark for http://purl.org/net/depmark#
```

Figure 4: A query asking the courses of all departments.

Yearbook The Yearbook system is in development and will contain a Web interface to query and navigate over the Yearbook information. One problem not completely solved in our prototype is a simple and flexible interface to present all the richness of the query language.

Discovery Cache The list of known Providers is kept in an RDF file which contains the URI where the actual local metadata of each component is.

Extensibility The plan is to extend this Yearbook to all chilean Departments. This is not an easy task, not only due to the diverse ontologies needed (for each area or type of Department), but also to the difficulty to introduce brand new technologies in other Departments besides Computer Science's.

3 Conclusions

Lessons from the prototype Building an ontology requires a deep knowledge of the domain to be modelled: something already known, but worth repeating. It was very useful in the process of building the ontology to have at hand typical examples of pages we wanted

to describe: first for creating the vocabulary and relations and second to test the "correctness and completeness" of the ontology.

One of the problems faced was the lack of tools to build the system, especially in the area of markup. This problem was already reported and studied [20], but the problem seems to be deeper. We found that there are not only problems in the process of markup or mantenaince of it, but also at the level of deciding at each stage if it is worth continuing with such process.

The existence of different roles of the contents show the existence of *spaces of markup*, that is, different views of the same contents which must be marked with different ontologies and kept in different spaces. This is an important remark for developers of markup tools, which should consider a feature to deal with this problem.

General conclusions We believe that an application of low complexity as a Yearbook allows testing principles of semantic integration on the Web without facing the problems usually found when using RDF in Web systems, particularly critical mass, markup tools and system development and maintenance [20]. Critical mass is not necessary but at the level of the organization. Due to the characteristic of a Yearbook, we can have trained people (one in each component) to do the markup process. The ontology development is basically a formalization of an informal ontology already used in practice and implicitly on documents. One of the main problems of Semantic Web applications, that of being very unfriendly, is in the Yearbook application not a problem: the system is designed to be built by few people in charge at each component. But the fruits will be harvested by all members of all components.

Summarizing, we presented an application which: Solves the creation-of-metadata bottleneck (by restricting to a particular domain of application); Extend functionality of current Yearbooks by making them interoperable from a semantic point of view; Suggests an approach to build catalogs with a discrete (periodic) temporal parameter; Allows a simple way of implementing massively historical memory. Finally, last but not least, will give the chilean academic community a reliable, rich and flexible way of querying the core organizations of our Universities: the Departments.

Acknowledgements Funded by Millenium Nucleous Center for Web Research, Grant P01-029-F, Mideplan, Chile.

References

[1] Henry Kim. Predicting how ontologies for the semantic web will evolve. *Communications of the ACM*, 45(2):48–54, February 2002.

[2] Fabien Gandon, Laurent Berthelot, and Rose Dieng-Kuntz. A multi-agent platform for a corporate semantic web. In C. Castelfranchi and W. Johnson, editors, *Interational Conference on Autonomous Agents, First International joint conference on Autonomous agents and multiagent systems*, pages 1025–1032, Bologna, Italy, July 2002.

[3] Stuart Madnick. The misguided silver bullet: What xml will and will not do to help information integration. In *Information Integration and Web-based Applications & Services (IIWAS2001)*, September 2001.

[4] A. Sheth. Changing focus on interoperability in information systems: From system, syntax, structure to semantic. In M. F. Goodchild, M. J. Egenhofer, R. Fegeas, and C.A. Kottman, editors, *Interoperating Geographic Information Systems*, pages 5–3, Kluwer, 1998. Academic Publishers.

[5] H. Wache, V. Ogele, T. Visser, U. Stuckenschmidt, H. Schuster, G. Neumann, and H. Ubner. Ontology-based integration of information - a survey of existing approaches. In H. Stuckenschmidt, editor, *IJCAI-01 Workshop: Ontologies and Information Sharing*, pages 108–117, September 2001.

[6] Rael Dornfest and Dan Brickley. The power of metadata. http://www.openp2p.com/pub/a/p2p/2001/01/18/metadata.html, 2001.

[7] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. Scientific American, May 2001.

[8] Varun Ratnakar and Yolanda Gil. A comparison of (semantic) markup languages. *Proceedings of the 15th International FLAIRS Conference, Special Track on Semantic Web*, May 2002.

[9] Peter Patel-Schneider and Jérôme Siméon. The yin/yang web: Xml syntax and rdf semantics. In *The Eleventh International World Wide Web Conference, WWW2002*, May 2002.

[10] Open directory project. http://dmoz.org/.

[11] Ora Lassila and Ralph Swick. Resource description framework (rdf) model and syntax specification. http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/, February 1999.

[12] Dublin core metadata initiative. http://purl.org/dc.

[13] Dan Brickley and R.V. Guha. Rdf vocabulary description language 1.0: Rdf schema. http://www.w3.org/TR/2002/WD-rdf-schema-20020430/, April 2002.

[14] Wolfgang Nejdl, Boris Wolf, Changtao QuLearning, Stefan Decker, Michael Sintek, Mikael Nilsson, Matthias Palmér, and Tore Risch. Edutella: A p2p networking infrastructure based on rdf. In *The Eleventh International World Wide Web Conference, WWW2002*, May 2002.

[15] N. F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson, and M. A. Musen. Creating semantic web contents with Protégé-2000. *IEEE Intelligent Systems Journal*, 16(2):60–71, 2001.

[16] Natalya Fridman Noy and Deborah L. McGuinnes. Ontology development 101: A guide to creating your first ontology. Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, Stanford Knowledge Systems Laboratory, March 2001.

[17] Ernesto Krsulovic and Claudio Gutierrez. Depmark, marcado con metadatos de departamentos universitarios chilenos. http://purl.org/net/depmark.

[18] Libby Miller. Inkling: Rdf query using squishql. http://swordfish.rdfweb.org/rdfquery/.

[19] Libby Miller, Andy Seaborne, and Alberto Reggiori. Three implementations of squishql, a simple rdf query language. In *1st International Semantic Web Conference (ISWC2002)*, Sardina, Italy, 2002.

[20] Stefan austein and Jörg Pleumann. Easing participation in the semantic web. In *Workshop at WWW2002, International Workshop on the Semantic Web*, May 2002.