VIVO Development Roadmap: Enhancing an Ontology-Based University Research Portal with OWL and Rules

Brian Lowe, Brian Caruso, and Jon Corson-Rikert

Cornell University, Albert R. Mann Library, Ithaca NY 14853, USA.
{bjl23,bdc34,jc55}@cornell.edu
http://mannlib.cornell.edu/

Abstract. VIVO (vivo.cornell.edu), a Virtual Life Sciences Library, is a production Web portal developed by Cornell University Library to provide integrated discovery of life sciences research activities taking place across Cornell's physical, administrative, and disciplinary divisions. The system comprises a custom RDBMS-based knowledge base store and a Java Web application driving the public portal as well as a Web-based ontology editing interface. The public site offers browsing and searching by instance type, hyperlinked traversal of object relationships, and some simple inferences to aggregate data across multiple object properties. A recent expansion in project scope to encompass scholarly activities across the entire university has prompted an investigation of how more extensive use of OWL and rule-based reasoning can leverage the structure of the knowledge base to serve more diverse query and display requirements.

1 Introduction

VIVO, a Virtual Life Sciences Library, is a service of Cornell University Library (CUL) developed in response to a major University initiative emphasizing the increasingly interdisciplinary nature of "New Life Sciences." CUL quickly recognized that Cornell's complex structure meant that discovering basic information about "who works on what where" was itself a formidable challenge.

VIVO addresses this issue by coupling a dynamic Web portal with an ontology-based model which allows for ready accommodation of new types of data in a flexible and open-ended structure. The ontology model describes such concepts as people, academic departments, laboratories and other facilities, publications, research grants and funding sources, and designated research focus areas. The class structure was based in part on the class hierarchy of the Advanced Knowledge Technologies (AKT) Support and Portal ontologies[1], but without explicitly importing AKT or committing to all of its semantics. Content panes in the public browsing interface can be automatically populated with instances of particular classes, and a full text search on names and datatype property values returns results grouped by type. When a user arrives at an instance display via browse or search, hyperlinks allow traversal of object property relationships to other

instance nodes, giving the user access to a web of content that might otherwise necessitate discovering and visiting multiple separate websites. VIVO was publicly launched early in 2005 and has since garnered significant positive feedback from throughout Cornell, spurring interest in leveraging the knowledge base for additional Web portals and an expansion to disciplines outside life sciences.

1.1 Current Technical Implementation

VIVO's primary purpose is to provide a practical service to researchers within and beyond Cornell as well as the general public, prompting the use of suitable technologies to support a public website. The VIVO knowledge base is currently stored in a MySQL 4.1 relational database, combining a general triple store model for object and datatype relationships with specialized tables for ontology structures. Of special importance is "sunrise" and "sunset" data attached to entities and relationships showing the time window during which the relevant item is asserted to exist, hold true, or be of relevance to the user. VIVO itself is a Java 1.5 Web application deployed on Apache Tomcat 5.5. JavaServer Pages (JSP) technology drives the Web page display layer, and Lucene 1.9 is used for full-text indexing and search functionality.

The semantics expressible through VIVO's ontology constructs currently align most closely with RDF Schema. A departure is the automatic inclusion of an inverse for each object property, as a core feature of the VIVO interface is bidirectional hyperlinking between entity nodes. In May 2007 the VIVO knowledge base consisted of about 200 classes, just over 200 properties, around 20,000 individuals, and roughly 200,000 instances of properties.

1.2 From manual curation to automated data integration

VIVO's database was initially populated largely with knowledge captured through a labor-intensive manual editing process in which human editors scoured various existing websites and publications for relevant content and relationships. The ontology thus far has been most useful for streamlining the input process by prompting or constraining assertions. Initial classification by human editors affects further property instantiation options.

The project, however, is now expanding in scope at a rate that precludes manual curation of all content. VIVO now automatically ingests publication information from several commercial databases. Research grant information is retrieved from a central administrative database and linked to principal investigators, funding agencies, and administering departments. In fall 2006 we processed an extract from Cornell's Human Resources database to expand the VIVO database's coverage of Cornell people to include at least basic data about all academic and professional staff. New Web portals under development will use these, and the current emphasis on integrating large amounts of data from external sources has prompted us to explore leveraging our ontology structure in new ways.

2 New Avenues for Development: OWL and Rules

As we supplement the manual curation process by integrating data from external sources, we gain additional explicit properties and relationships but ideal classification and filtering for display purposes becomes more challenging.

2.1 Applications for classification-type reasoning

The VIVO knowledge base currently supports multiple different Web portals, each of which displays a filtered subset of the available individuals. The most expedient way of implementing this functionality was to add special "flag" fields to the VIVO database schema, which must be set for each individual, often by human editors. This is a time-consuming and error-prone process, and the appropriate flag values for automatic additions to the knowledge base often cannot be reliably determined. A method for improving this situation is to employ defined OWL classes, which allows us to refashion these flags in a more robust manner. Instead of individually selecting all of the individuals that should appear in the College of Agriculture and Life Sciences (CALS) research portal, for example, we can create defined classes thusly:

$$Class(CALSUnit\ complete\ AcademicUnit \\ restriction(departmentOrDivisionWithin\ value(CALS)))$$

$$Class(CALSRelatedPerson\ complete\ Person \\ restriction(participantIn\ someValuesFrom(CALSUnit)))$$

The participantIn property subsumes official employment properties in addition to more informal collaborations, allowing such individuals as library liaisons to be classified as "CALS-related persons." We can then drive the portal filtering mechanism from the desired rdf:type values, rather than from the flag fields.

2.2 Applications for rules-based reasoning

There are several uses for special inferred properties, implemented either with simple SWRL rules or by taking advantage of the property chain construct provided by OWL 1.1. For example, graduate fields are associated directly with individual faculty members and not with official academic departments. But it is useful, from a department display, to be able to see the graduate fields with which its faculty members are associated:

$$hasFacultyMember(?x, ?y) \wedge memberOfGraduateField(?y, ?z) \rightarrow \\ associatedWithGraduateField(?x, ?z)$$

This technique is important in our application because we can infer from explicitly asserted properties a form for display that best matches the expectations of the user.

The *transitiveOver* axiom finds ready use in a number of situations. There is significant interest in using the VIVO knowledge base to discover knowledge about international research and outreach activities. It is powerful, for example, to make *conductsResearchIn* transitive over *partOf* and be able to readily infer that a faculty member conducting research in Mozambique is active in Africa.

2.3 Technical challenges and next steps

VIVO is a continually-updating knowledge base that needs to support manual editing with rapid visibility of changes, which places significant demands on the software used for reasoning. Initial proof-of-concept testing with Pellet[2] reasoner has been very promising, given its ability to handle incremental reasoning on ABox updates and support for OWL 1.1 semantics. We have also considered the OWLIM[3] Storage and Inference Layer (SAIL) for Sesame.[4] Though OWLIM is not a complete DL reasoner, its speed and memory footprint suggest it as a very practical method of adding basic inferencing to Web portals.

Current work with VIVO includes significant refactoring to support a variety of new applications of the underlying software both within and outside Cornell. A significant development challenge will be to seamlessly integrate persistent knowledge base storage with the desired inferencing capabilities. Meanwhile, the ontology used in the VIVO research portal is being tweaked to work as a proper extension of AKT, and there are a number of modeling decisions to be made surrounding the issues of faculty research keywords and possible extensions to the search mechanism to leverage mappings of alternative lexemes to controlled vocabulary or thesaurus terms. In all of these areas, the key will be to find the optimum balance of features and practical usability.

References

1. Advanced Knowledge Technologies: Support and Portal ontologies. `http://www.aktors.org/publications/ontology/`
2. Sirin, E. Parsia, B., Cuenca Grau, B. Kalyanpur, A., Katz, Y.: Pellet: A Practical OWL-DL Reasoner.
3. Kiryakov, A., Ognyanov, D., Manov, D.: OWLIM — a Pragmatic Semantic Repository for OWL. International Workshop on Scalable Semantic Web Knowledge (2005).
4. Sesame RDF Database. `http://openrdf.org/`