Position Paper
Pain Points and Patterns in Linked Enterprise Data

Alcatel-Lucent - Bell Labs

Alcatel-Lucent provides equipment and services to the telecommunications industry thereby enabling and facilitating the flow of data over enterprise networks and beyond. This data comes in many different forms and is usually highly distributed, making the development of applications that must integrate cross-source information very challenging. One area of research in Bell Labs involves the use of semantic web technologies to simplify the access to and integration of distributed heterogeneous data so as to facilitate application enablement. From this work we have identified a number of "pain points" encountered in trying to understand and facilitate the use of RDF for enterprise data, key among these being:

- poor support for dealing with dynamic data that changes over time
- issues encountered in processing large volumes of dynamic data
- relatively poor availability and adoption of standard ontologies with linked data
- primitive mechanisms for handling of non-binary relationships
- need for more powerful datatypes with automatic type casting and integrity checking
- poor coupling with existing tools/services (e.g. UML, RDBM, Web editors, Web Services)
- limited best practices and design patterns
- lack of mature and standardized APIs, particularly those that provide a more familiar object-oriented model

In this position paper we briefly elaborate on the first three of these.

Many enterprise applications depend on dynamic data; that is data that in a few minutes/hours/days might change (e.g., age, job title, employee count) or that may become invalid if you move to a new location (e.g., GPS coordinates, nearest gas station, local weather). RDF currently provides very poor support for managing information that might be modified or go out of date, particularly if you need to capture the temporal aspect of individual facts (i.e., triples). Two common patterns tend to be used for dealing with time-based data: time stamping and snap shots.

The time stamping pattern involves associating a time with every temporal fact. One way to achieve this is by reifying triples (not necessarily using RDF reification statements) and associating a datetime property value (e.g., using xsd:dateTimeStamp) with the blank node that now stands in for the fact. Another is to replace dynamic property values with blank nodes that have properties for the value and the timestamp (see [1] for the gist of this pattern). Both of these approaches then rule out the use of many of the inferencing capabilities inherent in RDFS and OWL (e.g., functional properties, transitivity, property chains). Alternatively, some triple stores (e.g. BigData[2]) allow triples to be stored with a fourth element of provenance that can be used to store a timestamp; annotated RDF and the query language AnQL [3] provide means for

annotating triples with timestamps. Unfortunately both of these approaches fall outside of current standards and there is no direct access to this provenance from within RDF/OWL axioms. In addition, most of methods (all except AnQL) require mechanisms beyond the capabilities of RDF/OWL reasoning to perform temporal reasoning on the timestamp facts like selecting the latest value of a property or determining whether a property had some value over a specific period of time.

The snap shot approach involves the separation of the data into distinct sub-graphs representing the facts that hold at specific points or periods in time. This approach affords the full use of the RDFS/OWL reasoning capabilities within individual time-slices but to perform temporal reasoning requires additional means for selecting and comparing values from multiple snapshots. Using SPARQL on a datastore where each snapshot lives in a separate named graph provides some support here in that queries can be constructed that span multiple graphs; but then additional support, such as that provided by (currently non-standard) SPIN [4], is needed if RDFS/OWL reasoning is required.

Whether time stamping or the taking of snap shots is employed there remains the need for formal temporal representations and semantics. The proposed Time Ontology [5] offers one solution but it has remained as a W3C Working Draft since it was first published in 2006.

The challenge of dealing with dynamic data is further complicated when the quantity of changing data is large. Techniques for handling large quantities of static RDF data have been improving rapidly for the past several years, but there is little support to handle problems that arise when parts of the data are highly dynamic. On large dynamic data sets, queries (with or without reasoning) can easily take long enough to process that the underlying data will have changed in the interim. Currently there is no way to discover that this change has happened aside from reissuing the query and comparing the results (and though even this is not guaranteed to always work). The general problem of managing and processing large-scale streaming RDF data remains an open challenge. If enterprises are to be convinced to move their real-time data streams to RDF this problem will have to be addressed by standards and practical tools and techniques.

If linked data is going to succeed within and across enterprises we will need to devote more attention to developing and standardizing enterprise ontologies. Simply using IRIs for resources, making them de-referencable through HTTP and linking between data sets (i.e., the requirements for Linked Data [6]) is not enough; marking two cross-source IRIs as owl:sameAs in no way guarantees they “mean” the same thing. Having common, formal ontologies to define business-critical concepts is the only way to ensure the meaning and interpretation of the data being shared between applications and services, whether in-house, business-to-business or business-to-customer. Enterprise-oriented ontologies for representing companies, organizations, processes, customers, etc are exist [7] and more are appearing [8] but their adoption has been slow and there has been no visible promotion of such ontologies from the

W3C. In so far as there is a pattern to follow here it is in the use of formal ontologies for assigning meaning to resources and defining common APIs.

1 Defining N-ary Relations on the Semantic Web. W3C Working Group Note 12, April 2006. http://www.w3.org/TR/swbp-n-aryRelations/
2 Big Data. http://www.systap.com/bigdata.htm
3 Nuno Lopes, Axel Polleres, Umberto Straccia, and Antoine Zimmermann. AnQL: SPARQLing up annotated RDFS. In Proceedings of the 9th International Semantic Web Conference (ISWC 2010), volume 6496 of Lecture Notes in Computer Science (LNCS), Shanghai, China, November 2010. Springer. http://iswc2010.semanticweb.org/pdf/51.pdf
4 SPIN: SPARQL Inferencing Notation, http://spinrdf.org/.
5 Time Ontology in OWL. W3C Working Draft 27 September 2006. http://www.w3.org/TR/owl-time/
6 Tim Berners-Lee, Linked Data, July 2006. http://www.w3.org/DesignIssues/LinkedData
7 Dietz, J.L.G. Enterprise Ontology - Theory and Methodology, Springer-Verlag Berlin Heidelberg, 2006.
8 Dieter Van Nuffel, Hans Mulder and Steven Van Kervel, Enhancing the Formal Foundations of BPMN by Enterprise Ontology. Lecture Notes in Business Information Processing, Volume 34, Part 3, 115-129, 2009.