Manageable Dataset Curation for Linked Data

Wei Tai[1], Kevin Feeney, Rob Brennan, Declan O'Sullivan

FAME & Knowledge and Data Engineering Group
School of Computer Science and Statistics
Trinity College Dublin, Dublin 2
Ireland
{WTai,Kevin.Feeney,Rob.Brennan,Declan.OSullivan}@scss.tcd.ie

This poster addresses the EKAW 2012 knowledge management and special focus areas by presenting the requirements and architecture for a manageable dataset curation tool designed to enable low overhead hosting of new public knowledge models. Our work contributes to the development of new methodologies and tools for knowledge management by presenting a knowledge administration process that reduces administrator effort while supporting distributed communities of administrators, authors and contributors. This is in contrast to most work to date on knowledge sharing that focus on easing the publication and consumption of the managed knowledge. At the heart of our architecture are components that translate, authorise and queue messy, real-world model update requests into SPARQL-Update queries that can leverage previous research on ontology evolution.

The success of Linked Data has led to an unprecedented rise in the number of structured datasets published on the web. However this success has highlighted the challenge of effectively curating the data and associated schemata or ontologies. Open linked data often relies upon domain-user-generated input and automated harvesting of third-party sources to build datasets. Thus published data tends to be 'dirty' [2] – with varying structure, limited compliance and frequent inconsistencies. From a dataset management perspective, it is often necessary that datasets should remain 'clean' for automated evolution support. However, when dealing with linked data, the maintenance overheads for attaining this can be prohibitive.

Ontology evolution management research has looked at the problem of reflecting changes in the underlying ontologies as well as dependent artefacts via a timely and systematic means so that ontologies remain consistent and up-to-date [1]. However these approaches cannot be directly applied to the challenge of effectively managing real-world 'dirty' data. Ontology reasoners cannot detect problems that are not defined as semantic inconsistencies in the ontology language [3]. Such problems are often a consequence of a failure to comply with domain or general rules, e.g. using an unasserted instance in property assertion. Although approaches have been developed to solve some problems of ontology evolution - such as capturing and representing change, ontology versioning, etc, they generally demand a lot of effort from the

[1] This work is partially funded by the SFI FAME Strategic Research Cluster (award No. 08/SRC/I1408) and the HEA NEMBES (PRTLI cycle 4).

dataset manager. In situations where the data set manager is a domain expert rather than an ontology expert – as is commonly the case with linked data – it is unrealistic to expect that they will successfully apply semantic approaches in the management of their datasets' evolution.

In this poster, we propose a framework which can support non ontology experts effectively managing datasets over time. This framework consists of an *input control* module, a *data store*, and a set of *viewing/publishing interfaces* for dataset access. Given that a lot of work has been done on accessing/viewing linked datasets through query languages [4] and through APIs, and RESTful interfaces, and also the wealth of established work on RDF data stores, the main focus of our work is on the input control module, which turns streams of dirty data from a wide range of sources into a clean semantic format that remains consistent over time, even as the schema evolves.

Our approach is based on the belief that the flexibility of RDF/RDFS is one of the major factors behind the complexity of data-set evolution management. By constraining the dataset to enforce a particular style of description, management complexity can be reduced without sacrificing descriptive power. Our inputs can come from diverse interfaces including Email and APIs. Once received, changes are checked against a set of constraints for validity in terms of system rules (e.g. all classes should have a label) or domain specific rules (e.g. a person should have a name). Failure to comply with constraints leads to warnings or errors. The semantic impacts of the changes are then analysed as in [1]. Rather than purely relying on users to choose from evolution strategies for each change at each resolution point, fixed evolution strategies can be defined. This sacrifices some flexibility but reduces the effort required by users to comprehend the different semantic impacts. Role-based access control and moderation queues are incorporated into our input control module. Any changes that break our constraints generate warnings or errors. Depending on the user's role, these cause the updates to be either rejected or forwarded for approval by moderators. Approved changes are transformed into SPARQL Update statements which can then be executed on the data store.

The poster provided will include requirements, an architecture and further details of the specific semantic constraints applied by the framework.

References

1. Maedche, A., Motik, B., Stojanovic, L., Stojanovic, N.: "User-driven Ontology Evolution Management," in Proc. of the International Conference on Knowledge Engineering and Knowledge Management, pp. 285-300 (2002)
2. Bizer, C., Heath, T., Berners-Lee, T.,: "Linked Data – The Story So Far," Int. J. Semantic Web. Inf. Syst. 5(3): pp. 1-22 (2009)
3. Motik, B., Grau, C. B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C.: "OWL 2 Web Ontology Language: Profile," in W3C Recommendation (2009)
4. Prud'hommeaux, E., Seaborne, A.: "SPARQL Query Language for RDF", in W3C Recommendation (2008)