Elsevier's Healthcare Knowledge Graph and the Case for Enterprise Level Linked Data Standards

Alex DeJong[1], Radmila Bord[1], Will Dowling[1], Rinke Hoekstra[1], Ryan Moquin[1], Charlie O[1], Mevan Samarasinghe[1], Paul Snyder[1], Craig Stanley[1], Anna Tordai[1], Michael Trefry[1], and Paul Groth[1]

[1] Elsevier
[2] Radarweg 29. Amsterdam 1043 NX
a.dejong, r.bord, w.dowling, p.groth, r.hoekstra, r.moquin, charlie.o, m.samarasinghe, p.snyder, c.stanley, a.tordai, m.trefry, @elsevier.com

Abstract. Linked Data principles and standards provide a powerful means for data integration across multiple sources. However, these standards are often too open to interpretation. In this work, we describe the necessity to define enterprise wide Linked Data standards in order to deliver Elsevier's healthcare knowledge graph and lessons learned in its implementation.

Introduction: Data from multiple parts of an organization is increasingly seen as an asset to enable new product functionality. This is similar at Elsevier. Elsevier provides a number of products in its health division that focus on different parts of the health system. For example, Order Sets provides groups of prepackaged actions for patients to do after diagnosis; Clinical Key provides key reference material for physicians, and the Gold Standard Drug Database provides comprehensive timely information about available medicines. As Elsevier looks to develop new products for clinical decision support across the clinical workflow[3], providing access to data from across products is necessary. Hence, our goal is to provide a data network that provides internal developers access to health data from different systems within Elsevier in a connected way.

However, there are a number of challenges in creating such a data network. First, the data is stored in different databases and the APIs differ across products. Second, each of these products has different pipelines for the sourcing and curation of its data. This includes already built applications that are designed for in-house experts and complex processes for data ingest. Lastly, the products are written on different software stacks, ranging from .NET to Java. Rewriting all these source systems on a common platform would be costly. Centralizing the data itself with a standard ETL process would introduce delays between data updates. Accessing the databases directly is also not advisable as, in many cases, there is coupling between application logic and the underlying database:

[3] https://www.elsevier.com/connect/how-information-analytics-can-help-researchers-and-clinicians

for example, knowledge about which tables can be combined, or about access and licensing controls is in the logic.

Thus, to tackle these challenges, we looked to Linked Data which addresses many of these concerns in terms of heterogeneity of technology stacks and the distributed nature of data. Here, we describe how we adapted Linked Data to Elsevier's context and a number of lessons learned in this internal adoption.

A Linked Data Approach: As a starting point, we asked three development teams to expose their data using APIs following the Linked Data principles.[4] In the course of initial development, it became clear that the breadth of technology choices that are Linked Data compliant is large and hindered interoperability. For example, end-points could provide RDF-XML as their serialization format or JSON-LD. Both are valid formats to establish an a Linked Data end-point, but a data consumer application that wants to use these end-points has to understand and implement two formats. Likewise, in other areas Linked Data is under specified. For example, security is is typically a limited consideration in the standards, since it is assumed all the data is publicly accessible on the Internet, which is true for many published endpoints but enterprise data needs to consider access requirements.

Thus, we took a two pronged approach. The first prong was, working with the development teams, to define a set of enterprise-wide Linked Data standards. Examples of constraints in these standards are as follows: All URIs need to be dereferenceable, support content-negotiation and at a minimum must return JSON-LD. A common set of namespace prefixes is specified. Class and property definitions need to follow a common set of naming conventions. Basic provenance information with a common set of provenance relations must be provided.

The second prong was to provide a central location (i.e. hostname) for access to the data network. This proxy maps selected context paths, e.g. /health/drugs, to a specific linked data API in the network. This central location has three benefits. It provides a well-known location to access Elsevier's Linked Data. It allows us to have control over what APIs are able to become officially part of the data network. Lastly, it allows us to employ standard security measures across the data network minimizing overhead on the decentralized providers. The combination of a technical enforcement point with clear guidance provided a path to implementation. Our current healthcare knowledge graph is currently being used in the development of a number of new products.

Conclusion & Lessons Learned: Overall, Linked Data has provided a viable mechanism for producing a cohesive data network. In the process, we came away with a number of key lessons learned: 1. an API centric approach is crucial as it fits in with standard web app models; 2. using existing developer tools (Jira, Git, Slack, Confluence) is important for integrating into the development cycle; 3. specifying which part of Linked Data technologies should be used is critical; 4. providing internal examples eases adoption; and 5. the platform agnostic nature of Linked Data is helpful in heterogenous environments within the enterprise.

[4] https://www.w3.org/DesignIssues/LinkedData