Chapter 2
Four Lessons in Versatility
or How Query Languages Adapt to the Web

François Bry, Tim Furche, Benedikt Linse, Alexander Pohl,
Antonius Weinzierl, and Olga Yestekhina

Institute for Informatics, University of Munich,
Oettingenstraße 67, D-80538 München, Germany
http://www.pms.ifi.lmu.de/

Abstract. Exposing not only human-centered information, but machine-processable data on the Web is one of the commonalities of recent Web trends. It has enabled a new kind of applications and businesses where the data is used in ways not foreseen by the data providers. Yet this exposition has fractured the Web into islands of data, each in different Web formats: Some providers choose XML, others RDF, again others JSON or OWL, for their data, even in similar domains. This fracturing stifles innovation as application builders have to cope not only with one Web stack (e.g., XML technology) but with several ones, each of considerable complexity.

With Xcerpt we have developed a rule- and pattern based query language that aims to give shield application builders from much of this complexity: In a single query language XML and RDF data can be accessed, processed, combined, and re-published. Though the need for combined access to XML and RDF data has been recognized in previous work (including the W3C’s GRDDL), our approach differs in four main aspects: (1) We provide a single language (rather than two separate or embedded languages), thus minimizing the conceptual overhead of dealing with disparate data formats. (2) Both the declarative (logic-based) and the operational semantics are unified in that they apply for querying XML and RDF in the same way. (3) We show that the resulting query language can be implemented reusing traditional database technology, if desirable. Nevertheless, we also give a unified evaluation approach based on interval labelings of graphs that is at least as fast as existing approaches for tree-shaped XML data, yet provides linear time and space querying also for many RDF graphs.

We believe that Web query languages are the right tool for declarative data access in Web applications and that Xcerpt is a significant step towards a more convenient, yet highly efficient data access in a “Web of Data”.

2.1 Introduction

The one undeniable trend in the development of the Web has been a move from human-centered information to more machine-processable data. This trend is a part of most visions for the future of the Web, may they be called “Web 2.0”, “Semantic Web”, “Web of Data”, “Linked Data”. There is a reason that this trend underlies so many of the visions for a future Web: With machine-processable data, other agents than the owner or publisher of data can create novel applications, e.g., by using the data in a
context never envisioned by the data owner, by presenting it in different ways or media, or by enhancing or mixing it with other data.

Unfortunately, though machine-processable data is called for by many of these visions, they do not agree on the data format. For human-centered information, HTML has clearly dominated the Web. For machine-processable data, Web 2.0 APIs and publishers tend to use XML, JSON, or YAML, Semantic Web publishers RDF and/or OWL. This way, application designers are either impeded from using data published in, say, RDF, if they are used to data in, say, XML or they have to cope with not only one (already fairly complex) stack of Web technologies but several.

The need for a more integrated, easier access to Web data has been recognized: For instance, the W3C has proposed a means of accessing XML data as RDF (GRDDL [54]). Other approaches integrate existing RDF query languages into XML query languages (XSPARQL [17], [29]) or vice versa ([60], SPAT [1]). In this work, we present a different answer to this problem: a single, unified language, called Xcerpt, that can query both XML and RDF with the same ease. Previous approaches require the user to learn (a) an XML (usually XPath or XQuery), (b) an RDF query language (usually SPARQL), and (c) how concepts from RDF and XML are mapped to each other, if at all. In our approach, we first develop a query language flexible enough to deal with most Web data (in the spirit of, though with quite different focus and result than [138]). Then we only have to teach the user how to query RDF resp. XML with that query language, reusing as much of the data and query concepts between the two settings as possible. Not only does this reduce the learning curve for the user considerably, it also makes it easy to extend the approach with further Web formats such as JSON, YAML, or Topic Maps.

We introduce Xcerpt in Sections 2.3.1 and 2.3.2 after a brief recall of the basics of the two Web formats considered here, XML and RDF, in Section 2.2.

But defining a language for unified access to XML and RDF is just how the story begins. For the approach to be feasible, we require two more ingredients: 1. a simple semantics that is nevertheless versatile enough to cover the specifics of both XML and RDF. 2. an evaluation engine that is competitive to engines specialized to XML or RDF data only.

In Sections 2.4.1 to 2.4.4 we propose two different ways to define the (declarative) semantics of Xcerpt: The first uses a modified form of simulation to describe which queries match what data. It is flexible enough to deal with queries on XML and RDF data and can be defined very concisely. We show in Section 2.4.2 and 2.4.3 how to adapt the (well-founded) semantics of rule programs with negation to use simulation rather than term equality/instantiation.

This gives an easy, straightforward definition of the semantics of Xcerpt. However, the disadvantage is that required notion of simulation is not as well studied as term equality and not supported by existing database or rule technologies. Therefore, we show in Section 2.4.4 how Xcerpt can be translated into standard Datalog with negation and value invention (Datalog_{new}) which can be evaluated by most SQL-database engines and many rule engines. Not only do we show how to translate Xcerpt into Datalog_{new}, but we do the same for XPath, XQuery, and SPARQL, thus establishing a uniform formal foundation for all these languages (that we exploit in Section 2.5.1 for

\[\text{http://www.w3.org/2007/01/SPAT/}\]