GRDDLing with Xcerpt: Learn one, get one free!*

François Bry Tim Furche Alina Hang Benedikt Linse

Institute for Informatics, University of Munich
Oettingenstrasse 67, 80538 Munich, Germany
http://www.pms.ifi.lmu.de/

1. INTRODUCTION

In the last years, the Semantic Web has significantly gained momentum, and the amount of RDF data on the Web has been increasing exponentially ever since the publication of the RDF recommendation. A great amount of this data is intermingled with HTML and XML at the help of microformats, embedded RDF and RDF/A.

To deal with this situation, the W3C has published an editor's draft soliciting a mechanism called GRDDL[1], an acronym for *Gleaning Resource Descriptions From Dialects of Languages* to the problem of extracting Semantic Web information from HTML and XML documents. The idea behind GRDLL is to associate an XML document containing embedded RDF information with one or more transformation programs – which the editor's draft proposes to write in XSLT. These transformation programs are specifically written to extract the RDF information from the document.

Moreover, the W3C also published a collection of descriptions of use-cases as a motivation for employing the GRDDL method. One of these use-cases is the scheduling of a meeting between friends who publish their calendars either as hcalendar, embedded RDF[2], RDFa or RSS 1.0 on their homepages. In a first step, the XSLT-stylesheets associated with the homepages are used for harvesting the RDF information from the homepages, and the resulting RDF graphs are combined in a single RDF model. In a second step, SPARQL is used to query the RDF data and find a date for the meeting that fits in everybody's schedule.

Our system implements this use case in two different manners. The first implementation follows the recommendation of the W3C editor's drafts, employing an XSLT processor and a SPARQL[3] implementation (in our case Jena[2]). The second implementation uses Xcerpt[4, 1], a versatile Web and Semantic Web query language for both processing stages. The implementation of these use-cases uncovers difficulties and challenges in the authoring of GRDDL algorithms in XSLT and also in Xcerpt, and highlights advantages and disadvantages of both approaches.

2. SYSTEM DESCRIPTION

hcalendar[3] is a calendaring and events format based on the iCalendar standard[4], which is used for embedding RDF data about calendars and events in arbitrary XML – but especially in HTML – documents. hcalendar data typically includes information about the title, description, start, end of an event and may also specify its duration and frequency.

Listing 1 shows a very simplistic application of the hcalendar format. In genuine HTML files enriched with hcalendar data found on the Web the usage of the vocabulary is more involved. In particular `vevent` tags may be nested within each other, and the data exhibits a very irregular structure. Therefore the task of retrieving the RDF triples from the files is a non-trivial one. Our system follows two complementary approaches for solving this use-case. In the first version, it uses XSLT to transform the HTML files into RDF/XML files only including the relevant RDF data. The resulting RDF/XML files are then loaded into a Jena RDF repository and SPARQL is used to schedule a meeting that fits the time constraints of all participants. In the second approach, Xcerpt is used both for the extraction of the RDF triples and for scheduling the meeting. Note that when compared with the second approach, the first one involves one redundant parsing step and one intermediate serialization.

Listing 1: Some sample hcalendar data

```
<html><head><title>hCalendar example</title></head>
  <body><span class="vevent">
      <h1 class="summary">Title of Event</h1>
      <p class="description">Description of an Event from
        <abbr class="dtstart" title="2007−10−18T09:00">
          18/10/2007 at 09:00 </abbr>until
        <abbr class="dtend" title="2007−10−18T10:00">
          10:00</abbr>
  </p></span><body></html>
```

2.1 GRDDLing with XSLT and SPARQL

Listing 2 shows a brief way of finding the description for an event whose XML-element is stored in the variable `$this_event`. A rather involved XPath query extracts the relevant text node which is then stored in an XSLT variable for reuse in the construction of the RDF/XML. To harvest complete RDF descriptions for hcalendar-embedded events, the same has to be done for their summaries, locations, start and end dates, etc.

Listing 2: Extraction of `hcal` information in the presence of nested events with XSLT

```
<xsl:variable name="description"
  select="descendant::∗[@class='description']
    [not(ancestor::∗[@class='vevent']
    [ancestor::∗[.=\$this_event]])]" />
```

Another challenge for the transformation from embedded hcalendar tags to RDF/XML is the conversion of the dates given in the format recommended by RFC 3339[5] to a structured representation of the date. Although this is not strictly necessary, it allows for a more human-friendly querying of the RDF data with SPARQL. Extracting the date and the time from a string such as `"2007-10-18T09:00"` is best achieved with the `xsl:anaylze-string`

[1] http://www.w3.org/2004/01/rdxh/spec

[2] http://research.talis.com/2005/erdf/wiki/Main/RdfInHtml

[3] http://microformats.org/wiki/hcalendar

[4] http://www.ietf.org/rfc/rfc2445.txt

[5] http://www.faqs.org/rfcs/rfc3339.html

command and a regular expression in XSLT. The code is ommitted here for the sake of brevity.

In accordance to the GRDDL use-case, our system uses the collected RDF data to find a date which fits into the schedules of all potential participants. SPARQL was designed to be a clean small language, which can do just enough in the context of querying RDF, without sacrificing its easy and efficient implementation.

Because of the tight restrictions (such as the lack of explicit negation and recursion) on the SPARQL language it is impossible to solve the use-case without generating a great amount of SPARQL queries with a scripting language or by generating an additional file including a finite set of dates that would come into consideration. The query in Listing 3 shows a SPARQL query which determines whether a given date is in conflict with other events within some RDF graph[6].

Listing 3: Checking for a conflicting event in an RDF calendar

```
Select ?title, ?x, ?y WHERE {
  ?x dc:title ?title. ?x cal:date ?y.
  ?y cal:startDay ?start. ?y cal:endDay ?end.
  ?y cal:startTime ?sTime. ?y cal:endTime ?eTime
  FILTER (
     ((?start = ?end && ?start = "2007-10-02" &&
       ?sTime <"12:00" && ?eTime > "11:00" )
     || (?start != ?end && ?start = "2007-10-02" &&
         ?sTime <"12:00")
     || (?start != ?end && ?end = "2007-10-02"
         && ?eTime > "11:00" )
     || (?start < "2007-10-02" && ?end > "2007-10-02"))
). }
```

2.2 GRDDLing with Xcerpt

Xcerpt is a pattern and rule based query language for semi-structured data in general and for XML and RDF in particular. Its *query patterns* feature a plethora of incompleteness constructs which allow to author brief and precise queries binding parts of the data to logical variables. *Construct terms* serve as templates to reassemble the bindings of the variables in a flexible manner. Construct and query terms are connected via rules which make up Xcerpt *programs* and can be evaluated both top down or bottom up. For the precise syntax and semantics of Xcerpt see [4].

Xcerpt is used to implement the entire GRDDL scheduling use case. Listing 4 gives an impression of how the extraction of calendar events from hcal-enriched HTML files is achieved with Xcerpt.

Listing 4: RDF harvesting with Xcerpt

```
CONSTRUCT
  rdf:RDF[ all cal:Event[
      cal:summary[ var Summary ],
      cal:descripttion[ var Description ], ... ] ]
FROM
  html {{ body {{
    desc /.*/((class="vevent")) {{
      optional desc (!/.*/((class="vevent")))*
        /.*/((class="summary")) { var Summary },
      optional desc (!/.*/((class="vevent")))*
        /.*/((class="description")) { var Description },
      ... }} }} }}
END
```

Querying the extracted RDF data with Xcerpt can be done in the same program, which saves the serialization of the RDF data in a file and the subsequent parsing, which is necessary for XSLT and SPARQL. It is achieved in a very similar way as the query above, and is omitted here for the sake of brevity. In contrast to the first solution of the use case, Xcerpt allows to recursively generate a list of possible dates (e.g. all dates lasting one hour and starting to the full hour between 8 a.m. and 8 p.m. in February). With this additional information it is indeed possible to automatically schedule a suitable date for the participants, and hence there is no need to embed the queries in a more powerful language.

3. NOVELTY OF THE APPROACH

Although GRDDL is still rated as a W3C editor's draft several implementations of the GRDDL mechanisms already exist, and also several of its use-cases have been implemented.

The Jena GRDLL reader is a GRDDL implementation for the Jena Semantic Web framework and automatically detects and applies stylesheets referenced within HTML pages for the purpose of extraction of RDF information. In contrast to our approach, it isn't an implementation of the use case itself, but of the GRDDL mechanism. Note that there are several other implementations of the GRDDL mechanism[7].

The W3C also published an online GRDDL demo[8], which allows to extract embedded FOAF, Creative Commons, RSS, Dublin Core and GeoURL data. In contrast to our approach, it does not deal with hcalendar data and it only implements the first step of a GRDDL use case.

Dan Conolly published an XSLT stylesheet[9] for extracting hcal information from XHTML. In contrast to our system, it does not deal with nested events, does not compare alternative ways for implementation, and again only deals with the first stage of the GRDDL use-cases.

4. SIGNIFICANCE OF THE APPROACH

Our system constitutes the first complete implementation of this GRDDL use-case and allows to draw the following conclusions:

Extracting RDF information from microformats is a non-trivial task and calls for expressive and user-friendly query languages specifically aimed at querying heterogeneous XML data.

For efficiency purposes it is desirable to have a language that is both capable of extracting the relevant information and of further semantic processing.

Although SPARQL is a very well-specified and probably the most famous RDF query language around, it lacks some features – such as recursion – which make its evaluation easier and less expensive, but delimit its expressiveness. In the context of GRDDL, these limitations mean that the SPARQL queries must be embedded in a more powerful general purpose programming language.

While Xcerpt is still a research prototype and not completely implemented, it already shows that versatile, pattern-oriented and rule-based querying has the potential to considerably ease the authoring of data intensive web-applications.

5. REFERENCES

[1] S. Berger, F. Bry, T. Furche, B. Linse, and A. Schroeder. Beyond xml and rdf: The versatile web query language xcerpt. In *World Wide Web Conference*, pages 1053–1054, 2006.

[2] B. McBride. Jena: A semantic web toolkit. *IEEE Internet Computing*, 6(6):55–59, 2002.

[3] E. Prud'hommeaux and A. Seaborne. SPARQL Query Language for RDF. Technical report, W3C, 2006.

[4] S. Schaffert. *Xcerpt: A Rule-Based Query and Transformation Language for the Web*. PhD thesis, University of Munich, 2004.

[6] `dc` and `rdf` are bound to the expected namespaces, the binding for `cal` is irrelevant

[7] `http://esw.w3.org/topic/GrddlImplementations`

[8] `http://www.w3.org/2003/11/rdf-in-xhtml-demo`

[9] `http://www.w3.org/2002/12/cal/glean-hcal.xsl`