Exercises and Projects

This project of medium difficulty can be done by two or three people in about two or three weeks. All required software is freely available. We provide some pointers to software that we have used successfully, but given the very active state of development of the field, the availability of software is likely to change rapidly. Also, if certain software is not mentioned, this does not indicate our disapproval of it.

The assignment consists of three parts.

1. In the first part, you will create an ontology that describes the domain and contains the information needed by your own application. You will use the terms defined in the ontology to describe concrete data. In this step you will be applying the methodology for ontology construction outlined in the first part of this chapter and using OWL as a representation language for your ontology (see chapter 4).

2. In the second part, you will use your ontology to construct different views on your data, and you will query the ontology and the data to extract information needed for each view. In this part, you will be applying RDF storage and querying facilities (see chapter 2).

3. In the third part, you will create different graphic presentations of the extracted data using web-based technology.

Part I. Creating an Ontology

As a first step, decide on an application domain, preferably one in which you have sufficient knowledge or for which you have easy access to an expert with that knowledge.

In this description of the project, we use the domain of radio and television broadcasting, with its programs, broadcasting schedules, channels, genres, and celebrities. Of course, you can replace this with any domain of your choosing. In our own courses,

we use different domains every year ranging from university ranking to movies and the Fortune 500.

Second, you will build an ontology expressed in OWL that describes the domain (for example, your faculty). The ontology does not have to cover the whole domain but should contain at least a few dozen classes. Pay special attention to the quality (breadth, depth) of the ontology, and aim to use as much of OWL's expressiveness as possible. There are a number of possible tools to use at this stage. Arguably the best current editor is Protégé,[28] but we have also had good experiences with TopBraid Composer.[29]

If you are ambitious, you may even want to start your ontology development by using ontology extraction tools from text or experimenting with tools that allow you to import semistructured data sources, such as Excel sheets or tab-delimited files (see, for example, Excel2RDF, ConvertToRDF, Any23, or XLWrap). Of course, you may choose to start from some existing ontologies in this area.

Preferably, also use an inference engine to validate your ontology and to check it for inconsistencies. If you use Protégé, you may want to exploit some of the available plug-ins for this editor, such as multiple visualizations for your ontology or reasoning with Pellet or HermiT.

Third, populate your ontology with concrete instances and their properties. Depending on the choice of editing tool, this can either be done with the same tool (Protégé) or, given the simple syntactic structure of instances in RDF, you may even decide to write these by hand or to code some simple scripts to extract the instance information from available sources. For example, you can convert a relational database with the given data to RDF. Or you may want to write a scraper for some of the many websites that contain information on radio and television schedules, programs, genres, and celebrities. The BBC even offers a convenient application programming interface for querying their schedule directly.[30] You may want to use the syntax validation service

[28] http://protege.stanford.edu/.
[29] See http://topquadrant.com/products/TB_Composer.html.
[30] See http://www.bbc.co.uk/programmes/developers.

offered by W3C[31] – this service not only validates your files for syntactic correctness but also provides a visualization of the existing triples. Also, at this stage, you may be able to experiment with some of the tools that allow you to import data from semistructured sources.

At the end of this step, you should be able to produce the following:

- The full OWL ontology
- Instances of the ontology, described in RDF
- A report describing the scope of the ontology and the main design decisions you made while modeling it.

Part II. Profile Building with SPARQL Queries

Here you will use query facilities to extract relevant parts of your ontology and data. For this you need some way of storing your ontology in a repository that supports both query and reasoning facilities. You may use the Sesame RDF storage and query facility,[32] which comes bundled with an OWLIM reasoner. We have also found that the Joseki Sparql Server is a nice starting point as it provides a built-in web server.

The first step is to upload your ontology (as RDF/XML or Turtle) and associated instances to the repository. This may involve some installation effort.

Next, use the SPARQL query language to define different user profiles, and use queries to extract the data relevant for each profile.

In the example of modeling television programs, you may choose to define viewing guides for people with particular preferences (sports, current affairs) or viewers of particular age groups (e.g., minors), to collect data from multiple television stations (even across nations), to produce presentations for access over broadband or slower mobile connections, and so on.

[31] www.w3.org/RDF/Validator/.
[32] www.openrdf.org/.

The output of the queries that define a profile will typically be in XML or JSON (JavaScript Object Notation).

Part III. Presenting Profile-Based Information

Use the output of the queries from part II to generate a human-readable presentation of the different profiles. There exist several convenient libraries for querying a SPARQL endpoint from your favorite programming language: Python has SPARQLWrapper,[33] PHP has the ARC2 library,[34] and Java users will like ARQ, which is part of the popular Jena library.[35] There are even libraries for visualizing SPARQL results from Javascript, such as sgvizler.[36]

The challenge of this part is to define browsable, highly interlinked presentations of the data that were generated and selected in parts I and II.

Alternative Choice of Domain

Besides using the semistructured dataset describing the broadcasting domain, it is possible to model the domain of a university faculty, with its teachers, courses, and departments. In that case, you can use online sources, such as information from the faculty's phonebook, curriculum descriptions, teaching schedules, and so on to scrape both ontology and instance data. Example profiles for this domain could be profiles for students from different years, profiles for students from abroad, profiles for students and teachers, and so on.

Conclusion

After you have finished all parts of this project, you will effectively have implemented large parts of the architecture shown in figure 7.1. You will have used most of the

[33] http://sparql-wrapper.sourceforge.net/.
[34] http://incubator.apache.org/jena/documentation/query/index.html.
[35] http://incubator.apache.org/jena/.
[36] code.google.com/p/sgvizler/.

languages described in this book (RDF, RDF Schema, SPARQL, OWL2), and you will have built a genuine Semantic Web application: modeling a part of the world in an ontology, using querying to define user-specific views on this ontology, and using web technology to define browsable presentations of such user-specific views.