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/.