Interoperation between Information Spaces on the Web

Andreas Harth
andreas.harth@deri.org
Digital Enterprise Research Institute
National University of Ireland, Galway

December 11, 2005

Abstract

In my thesis I will address the problem of interoperation between information spaces on the web. We explain how this problem is different to traditional database integration scenarios. In particular, we focus on one issue of the information integration problem peculiar to the web environment, namely linking information across sources. We use a graph-based data model and representation format, and define a query and rule language where queries can span multiple sources. We show how this language can be used to relate information between sources.

1 Introduction

In recent years, there has been a transition from a traditional view in data integration systems with a hierarchical architecture towards a completely distributed model associated with peer-to-peer (p2p) systems [5, 1, 9, 3, 7]. Rather than integrating a set of disparate information sources into a single global schema residing at one server, the p2p view assumes a complex system comprising a large number of autonomous sources that relate information among themselves.

The p2p scenario shares many of the characteristics of the web: web servers are autonomous sources providing documents that are related to each other using hyperlinks. Similarly, the web is a self-organizing system in a sense that its global structure emerges from and evolves by collective creation and linkage of HTML pages, without the need for a central server or coordination authority.[1] In addition, the web community has developed a number of languages that aim at representing semistructured data or ontologies and thus facilitate the exchange of information.

Although there are many similarities, the web environment differs from the typical p2p data integration scenario in the following ways:

- the network structure on the web is relatively stable; servers are up most of the time, in contrast to peers joining very frequently
- relating information on the web is done manually; the network structure is not emerging randomly as assumed by the p2p model, but *users* put in hyperlinks to connect related pages
- on the web, identifiers and their meaning may be shared across sources; databases identifiers are local and may mean different things in different databases

We believe it is not desirable to completely mechanize all integration activities for data on the web. In particular, associations that relate information across sources should be crafted manually. In fact, the manually created hyperlink structure of the HTML web provides a rich source of information which is useful e.g., for ranking purposes. We believe that a typical system for information integration and sharing on the web can comprise at least the following components:

1. **Query and reasoning.** Infrastructure that allows to execute queries which take into account the link structure between sources; in particular recursive query processing facilities are needed because the network structure can contain cycles.

[1] except maybe W3C which standardizes languages, formats and protocols to provide interoperability

2. **Link creation and exchange.** Users may create associations between data in the form of coordination rules, and store, publish, and exchange these coordination rules with other people.

3. **Information browsing.** User interfaces are needed to browse and explore the integrated information set.

4. **Caching and replication.** Caching and replication mechanisms are required at some stage to make the system fast, scalable, and reliable.

In the thesis, we focus on the problem discussed in point 1, since we believe that has to be solved before the other components become useful or are required.

To motivate the problem, consider three autonomous data sources with connections and dependencies among them as illustrated in Figure 1: one data source contains information about people expressed in FOAF[2] vocabulary collected from the web, another data source holds publication metadata from DBLP[3], and the third data source contains Citeseer[4] publication data. Since the data sources contain overlapping or complementary information, we can use the following links to relate data in one source to data in another source:

- information about people from FOAF files should be also made available in the DBLP data source, e.g., the DBLP admin wants to show a picture of the authors of a publication
- publications in the DBLP source should also include abstracts which are available in the Citeseer data set
- the administrator of the FOAF source wants to show information about the publications of a person

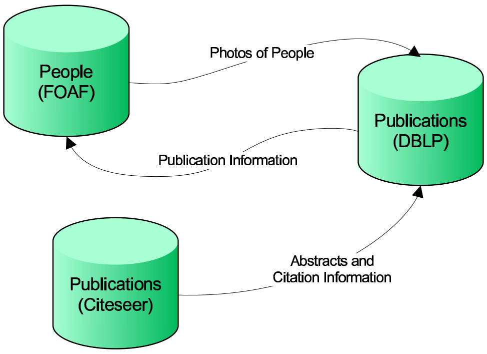

Figure 1: Three data sources connected via coordination links.

We assume that all data is available in RDF (Resource Description Framework), and sources use Dublin Core[5] vocabulary for describing publications and FOAF vocabulary for describing persons.

The research question this thesis aims to answer is two-fold:

- How to link RDF data sources on the web to arrive at a distributed, self-organized system comprising a large number of autonomous interoperable information sources?
- How to utilize these links to interoperate (share, exchange, and integrate information) among the connected sources?

The remainder of the paper is organized as follows. In Section 2 we introduce RDF with context, the data model we assume for our framework. Section 3 introduces syntax and semantics of our query and rule language, based on Notation3 syntax. Section 4 discusses the current state of the system and presents some ideas for future work, and Section 5 concludes.

[2] http://www.foaf-project.org/

[3] http://www.informatik.uni-trier.de/~ley/db/

[4] http://citeseer.ist.psu.edu/

[5] http://www.dublincore.org/

2 Data Model

In this section we describe RDF, the data model used, and introduce our notion of context, which is mandatory in a distributed environment. RDF is a schema-less, self-describing graph-based data model, which facilitates merging information from different sources without a need for schema integration at the merging stage.

2.1 RDF

We begin with defining the standard RDF data model, which consists of RDF triples.

Definition 2.1 (RDF Triple) *Given a set of URI references $\mathcal{U}$, a set of blank nodes $\mathcal{B}$, and a set of literals $\mathcal{L}$, a triple $(s, p, o) \in (\mathcal{U} \cup \mathcal{B}) \times \mathcal{U} \times (\mathcal{U} \cup \mathcal{B} \cup \mathcal{L})$ is called an RDF triple.*

In such a triple, s is called the subject, p the predicate, and o the object. We refer the interested reader to various W3C Recommendations [14, 12] which describe the RDF model in more detail, including blank nodes, containers, and reification. We use Notation3 (N3) as a syntax for RDF. For a full description of N3 see [4].

To make this paper self-contained, we introduce the basic syntactic N3 primitives. Brackets (<>) denote URIs, quotes (””) denote RDF literals, and blank node identifiers start with “_ :”. There exists a number of syntactic shortcuts, for example “;” to introduce another predicate and object for the same subject. Namespaces can be introduced with the @prefix keyword.

Figure 2 shows a small example in N3 syntax describing a paper and a person.

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<http://www.informatik.uni-trier.de/~ley/db/journals/computer/computer25.html#Wiederhold92>
         dc:title "Mediators in the Architecture of Future Information Systems." ;
         dc:creator <http://www.example.org/dblp/GioWiederhold> ;
         rdf:type foaf:Document .

<http://www.example.org/dblp/GioWiederhold> foaf:name "Gio Wiederhold";
         foaf:homepage <http://www-db.stanford.edu/people/gio.html> ;
         rdf:type foaf:Person .
```

Figure 2: Simple N3 example describing a document and a person.

RDF has the notion of object identity via URIs and object nesting via predicates which refer to another URI. More advanced object-oriented features such as classes and inheritance can be layered on top of the simple graph-based data model of RDF.

2.2 Context

Although the RDF specification itself does not define the notion of context, usually applications require context to store various kinds of metadata for a given set of RDF triples. In typical integration scenarios where data is gathered from large number of sources, it is mandatory to track the provenance of information, i.e. the physical location of the RDF file addressable via URI. Capturing provenance is one of the fundamental necessities in open distributed environments like the web, where the quality of data has to be judged by its origin.

The interpretation of context depends on the application. For example, in an information integration use case, the context is the URI of the file or repository from which a triple originated. Contexts are useful in other application scenarios as well, such as versioning or access control.

In the following we define our simple notion of context.

Definition 2.2 (Triple in Context) *A pair (t, c) with t be a triple and c $\in (\mathcal{U} \cup \mathcal{B})$ is called a triple in context c.*

Please note that a pair ((s, p, o), c) in context c is equivalent to the quadruple (s, p, o, c). In our model, we assume a finite set of data sources which are accessible via HTTP. Each data source can host multiple contexts. A context c can be a relative URI or an absolute URI. A relative URI denotes a context relative to the current context, which allows to move the location of entire data sets while keeping the internal link structure intact.

Example 2.1 *For our running example, the context for data from DBLP is http://example.org/dblp, for FOAF http://example.com/foaf, and for Citeseer http://example.org/citeseer. For brevity, we will use exo:dblp, exc:foaf, and exo:citeseer to denote the data sources in the rest of the paper.*

Notation3 has an extension which allows quoting of graphs. Within N3, RDF subgraphs can become the subject or object of a statement, using "{}". To be able to express our notion of context within the RDF data model we introduce the namespace `yars`[6]. The `yars:context` predicate denotes that the subgraph grouped in the subject occurs in the context provided as the object.

2.3 Related Approaches

The relational data model is by far the most popular, but has some drawbacks when it comes to data exchange and integration. In particular, data cannot be exchanged before an agreement is reached on how different relational schemas relate to each other.

Semistructured data formats such as OEM [15] or XML try to alleviate some of the problems because data in these formats can be merged without the need to integrate the schema first. However, one drawback of XML is the lack of the notion of objects and object identity. In RDF, the concept of URI acts as a global identifier for entities, which represents a form of agreement among multiple sources about how to name things. In addition, ontologies layered on top of RDF can be used to describe the semantics of properties in a formal way which allows to reason over these formal descriptions.

We extend RDF with a notion of context which we believe is mandatory in the distributed web environment. Our notion of context is relatively basic in a sense that we only capture the physical location of a given piece of information, not the processing steps performed to derive the information (e.g. data exchange, joining information from multiple sources) which is addressed by more complex provenance tracking frameworks [6, 8, 18].

3 Query and Rule Language

Rules are used to pose queries against a set of information spaces, or to specify how pieces of information are related to each other. In the following, we define syntax and semantics of our rule language.

3.1 Rule Syntax

To be able to express rules, N3 extends the RDF data model with variables. Variables in N3 are denoted using a question mark "?" as prefix. With variables, we can define the notion of quad patterns, which allow us to specify patterns where constants may be replaced by variables.

Definition 3.1 (Quad Pattern) *Given a set of variables $\mathcal{V}$, a quad (s, p, o, c) $\in (\mathcal{U} \cup \mathcal{B} \cup \mathcal{V}) \times (\mathcal{U} \cup \mathcal{V}) \times (\mathcal{U} \cup \mathcal{B} \cup \mathcal{L} \cup \mathcal{V}) \times (\mathcal{U} \cup \mathcal{B})$ is called an RDF quad pattern.*

If the context position is not specified, we assume as the context the location (URI) where the set of triples is stored. To be able to formulate rules within the RDF data model we introduce the `log:implies` predicate in namespace `log`[7]. We again use the graph quoting feature of N3.

Definition 3.2 (Rule) *A rule is in the form $\{B_1 \dots B_n\}$ log:implies $\{H\}$. where $B_1 \dots B_n$ and H are quad patterns.*

[6] `http://sw.deri.org/2004/06/yars#`

[7] `http://www.w3.org/2000/10/swap/log#`

The quad patterns $B_1 \ldots B_n$ are called the body, and H the head of a rule. A rule is safe if all variables occurring in the head also occur in the body. Please note that rules itself have an associated context, that is, where the rule is accessible on the web.

Example 3.1 *Recall the running example given in the introduction. We are now able to give an example of a rule that expresses a link between repositories (we assume definitions of the namespaces* `foaf, yars, log`*). The following rule is stored at exo:dblp and relates photos in exc:foaf to people in exo:dblp by matching on a common homepage property.*

```
{{ ?x foaf:homepage ?h .
   ?x ?p ?o . } yars:context exc:foaf .
   ?y foaf:homepage ?h .
} log:implies {
   ?y ?p ?o .
} .
```

3.2 Operational Semantics

We have defined the syntax of rules, but not yet their meaning. In the following, we define an operator that can be used to calculate a fixpoint taking into account possibly recursive rules.

Definition 3.3 (Immediate Consequence) *Let P be a program at context C_P. A fact A is an immediate consequence of P if (i) $A \in C_P$ or (ii) $\{B_1 \ldots B_n\}$ log:implies $\{A\}$. is an instantiation of a rule in P and each $B_i \in C_i$. T_P denotes all facts that are immediate consequences of P.*

In our framework there are multiple rules, possibly recursively referencing each other. There is no way to escape the need for recursion, since there is no central control of the network, and each actor can put in rules without coordinating with others, which makes it possible (and very likely) that one actor references triples inferred by a rule of another actor. In addition, to be able to define the semantics of transitivity we need recursion.

Given that our rules language has no function symbols that can be used to generate new symbols, the set of rules are guaranteed to reach the fixpoint (i.e. no new facts are generated by the T_P operator) after a finite number of steps [2].

3.3 Related Approaches

Unlink traditional data integration approaches, we do not need to integrate a schema, since RDF is semistructured and schema-less. We only need to relate identifiers to each other, similar to what is done in [13]. Ideally, schema-level identifiers, that is, URIs that identify properties and classes, are commonly used across information spaces, so there is no need to actually map schema-level information. We believe that rule unfolding is a sufficient method for our representation format that is somewhat between full schema mapping and merely relating identifiers to each other.

SPARQL is a query language for RDF[8] that shares many of the features of our framework. The most notable distinctions between our approach based on N3 and SPARQL are: 1) we use N3 syntax to encode both facts and rules, which allows to write rules that return other rules as result of a query, 2) we allow recursion in our language, 3) the notion of context in our framework is more involved than in SPARQL.

4 Discussion and Ongoing Work

We have implemented a prototype that allows to answer conjunctive queries that span multiple contexts. We currently experiment with an RDF version of DBLP (around 1.5 GB in size) and a web crawl of RDF data (around 1 GB in size). [10] has more information on the index organization and query processing used in our prototype. One immediate next step is to implement the operational semantics described in Section 3.2. We plan to also investigate the use of (hybrid) top-down methods such as QRGT [17].

[8] currently a W3C Working Draft

One of the major challenges we acknowledge but only partially tackled in our framework is that of object identity. Especially, how to handle objects that have different identifiers but denote the same real-world entity? Ontology languages such as OWL have means to describe "inverse functional properties" of objects, that is, objects that share the same value for a property denote the same real world entity. We can specify these properties in the mapping rules, however, inverse functional properties detected during the query processing phase are not taken into account. What is needed is an extension in the query processing component to be able to perform multiple rounds during query evaluation, where in each round new information is taken into account which has been obtained in the previous round.

A potential optimization to the query processing is to infer new links that can act as shortcut between sources. E.g., assume a rule on A that references context B which then references context C. Rather than routing a query via B all the time, the system could assume a shortcut and directly link A and C, taking out the unnecessary step via B.

Ongoing work includes adding rules with scoped negation [16], a form of negation-as-failure, and investigating how the semantics of ontology languages such as RDFS and OWL DLP [11] can be layered on top of our framework. An interesting theoretical question which requires some more in-depth investigation is what extensions (possibly within second-order logic) are needed to be able to query rule bases.

5 Conclusion

The aim of the thesis is to investigate methods for interoperation among autonomous RDF data sources across the web. Our framework allows to operate with only local information, that is, creating links between data source or evaluating queries can be done without any global knowledge about the network.

We have shown how to use coordination links, expressed in the query and rules language Notation3 with context, to interlink data sources. We have defined both syntax and the operational fixpoint semantics of the query and rules language, and sketched a number of questions we intend to tackle as part of the Ph.D. thesis.

References

[1] K. Aberer, M. Punceva, M. Hauswirth, and R. Schmidt. Improving data access in p2p systems. *IEEE Internet Computing*, January/February 2002.

[2] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases.* Addison-Wesley Publishing Co., 1995.

[3] M. Arenas, V. Kantere, A. Kementsietsidis, I. Kiringa, R. J. Miller, and J. Mylopoulos. The hyperion project: From data integration to data coordination. *SIGMOD Record*, 32, 2003.

[4] T. Berners-Lee. Notation 3 – ideas about web architecture. http://www.w3.org/DesignIssues/Notation3.html.

[5] P. A. Bernstein, F. Giunchiglia, A. Kementsietsidis, J. Mylopoulos, L. Serafini, and I. Zaihrayeu. Data management for peer-to-peer computing: A vision. In *Workshop on the Web and Databases, WebDB 2002*, 2002.

[6] P. P. de Siliva and D. McGuiness. Knowledge provenance infrastructure. *IEEE Data Engineering Bulletin*, 26(4):26–32, 2003.

[7] E. Franconi, G. Kuper, A. Lopatenko, and I. Zaihrayeu. The codb robust peer-to-peer database system. In *Proc. of the 2nd Workshop on Semantics in Peer-to-Peer and Grid Computing (SemPGrid'04)*, 2004.

[8] R. V. Guha, R. McCool, and R. Fikes. Contexts for the semantic web. In *Proceedings of the 3rd International Semantic Web Conference, Hiroshima*, Nov. 2004.

[9] A. Y. Halevy, Z. G. Ives, P. Mork, and I. Tatarinov. Piazza: Data management infrastructure for semantic web applications. In *Proceedings of the 12th International WWW Conference*. ACM Press, 2003.

[10] A. Harth and S. Decker. Optimized index structures for querying rdf from the web. In *Proceedings of the 3rd Latin American Web Congress*. IEEE Press, 2005.

[11] A. Harth and S. Decker. Owl lite- reasoning with rules, 2005. WSML Working Draft.

[12] P. Hayes. Rdf semantics. W3C Recommendation, Feb. 2004. http://www.w3.org/TR/rdf-mt/.

[13] A. Kementsietsidis, M. Arenas, and R. J. Miller. Managing data mappings in the hyperion project. In *Proceedings of the 19th International Conference on Data Engineering (ICDE)*, 2003.

[14] F. Manola and E. Miller. Rdf primer. W3C Recommendation, Feb. 2004. http://www.w3.org/TR/rdf-primer/.

[15] Y. Papakonstantinou, H. Garcia-Molina, and J. Widom. Object exchange across heterogeneous information sources. In *Proceedings of the 11th International Conference on Data Engineering (ICDE)*, pages 251–260, 1995.

[16] A. Polleres, C. Feier, and A. Harth. Rules with contextually scoped negation. Submitted.

[17] J. D. Ullman. *Principles of Database and Knowledge-Base Systems*, volume 2: The New Technologies. Computer Science Press, 1989.

[18] Y. Velegrakis, R. J. Miller, and J. Mylopoulos. Representing and querying data transformations. In *Proceedings of the 21st International Conference on Data Engineering (ICDE)*, 2005.