

Interoperation between Information Spaces on the Web

Andreas Harth
andreas.harth@deri.org

Digital Enterprise Research Institute
National University of Ireland, Galway

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 such a language can be used to relate information among 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 [6], [1], [13], [3], [10]. 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 recently 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 and leaving 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

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

- 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 assume it is neither desirable nor possible to completely mechanize all integration activities for data on the web. 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.
2. **Link creation and exchange.** Agents (users or software) may create associations between data in the form of coordination rules, and store, publish, and exchange these coordination rules with other agents.
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 problem has to be solved before the other components become useful or are required. We assume that all data is available in RDF (Resource Description Framework); data in other formats can be aligned using wrappers.

Example *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 FOAF² vocabulary collected from the web, another data source holds publication metadata from DBLP³, and the third data source contains Citeseer⁴ 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*

² <http://www.foaf-project.org/>

³ <http://www.informatik.uni-trier.de/~ley/db/>

⁴ <http://citeseer.ist.psu.edu/>

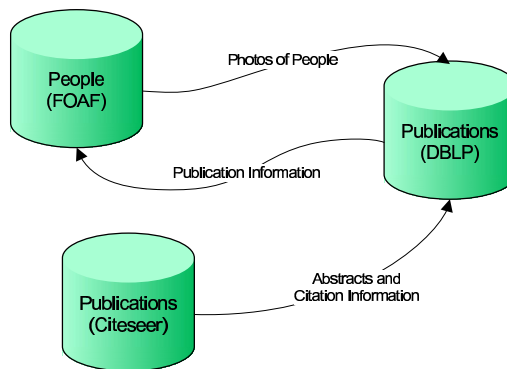


Fig. 1. Three data sources connected via coordination links.

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 the data model for our framework. Section 3 introduces syntax and semantics of the query and rule language. In Section 4 we describe how to encode and integrate the formal semantics of vocabulary descriptions into the framework. Section 5 discusses the current state of the system and presents some ideas for future work, and Section 6 concludes.

2 Data Model

In this section we first review traditional data models, then describe RDF, the data model we use, and finally introduce our notion of context, which is mandatory in a distributed environment.

2.1 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 [19] or XML try to alleviate some of the problems because data in these formats can be merged without the need for integrating the schema first. However, one drawback of XML is the lack of the notion of objects and object identity.

We employ the notion of context which we believe is mandatory in the distributed web environment. Context frameworks such as [9], [12], [23] track the processing steps performed (e.g. data exchange, joining information from multiple sources) to derive the associated piece of information. TRIPLE [21] has the notion of parameterized contexts, where parameters can be passed to a set of facts and rules. Our notion of context is relatively basic in a sense that we just capture the physical location of a given piece of information.

2.2 RDF Data Model

Before we describe our notion of context, we define the standard RDF data model. RDF is a schema-less, self-describing graph-based data model, which facilitates merging information from different sources without performing schema integration at the merging stage. The data model consists of RDF triples.

Definition 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 [17] which describes the RDF model in more detail, including blank nodes, containers, and reification.

We use N3 (Notation3) as a syntax for RDF. To make this paper self-contained, we introduce the basic syntactic N3 primitives. Brackets ($\langle \rangle$) denote URIs, quotes (" '' ") denote RDF literals, and blank node identifiers start with " $_ \text{:}$ ". There exists a number of syntactic shortcuts, for example " ; " to introduce another predicate and object for the same subject. Namespaces are introduced with the @ prefix keyword. For a full description see [5]. Figure 2 shows a small example in N3 syntax describing a paper and a person.

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. RDF has a notion of object identity via URIs and object nesting via predicates which refer to other URIs. More advanced object-oriented features such as classes and inheritance can be layered on top of the simple graph-based data model of RDF, which is discussed in Section 4.

2.3 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 a large number of sources, it is mandatory to track the provenance of

```

@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#> .
@prefix ley: <http://www.informatik.uni-trier.de/~ley/> .

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 .

```

Fig. 2. Small N3 example describing a document and a person.

information, that is, the physical location of the RDF file addressable via a 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.

In the following we define our basic notion of context.

Definition 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 $((s, p, o), c)$ is equivalent to a quadruple (s, p, o, c) . In our model, we assume a finite set of spaces which are accessible via HTTP. Each information space 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 *For our running example, the context for information 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.*

N3 extends the RDF data model with means to quote 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`⁵. The `yars:context` predicate denotes that the subgraph grouped in the subject occurs in the context provided as the object.

⁵ <http://sw.deri.org/2004/06/yars#>

3 Query and Rule Language

Rules are used to pose queries, or to specify how pieces of information, possibly in different contexts, are related to each other. We first review some related approaches, and then define syntax and semantics of our rule language.

3.1 Related Approaches

Unlike traditional data integration approaches, we can postpone the schema integration to a later stage, since RDF is semistructured and schema-less. We only need to relate identifiers to each other, similar to what is done in [16]. We believe that rule unfolding is a sufficient evaluation method that is somewhat between full schema mapping and merely relating identifiers to each other.

Many query and rule languages for RDF have been proposed [4]. We discuss here only SPARQL⁶ due to space limitations. SPARQL is a query language for RDF that shares many of the features of our framework. The most notable distinctions between our approach based on N3 and SPARQL are that we use N3 syntax to encode both facts and rules, which can be used to write rules that return other rules, and we allow recursion in our language.

3.2 Notation3 Rule Syntax

To be able to express rules, N3 extends the RDF data model with variables. Variables in N3 are prefixed with a question mark (“?”). 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 (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.*

To be able to formulate rules within the RDF data model we use the graph quoting feature of N3 and introduce the `log:implies` predicate in namespace `log`⁷.

Definition 4 (Rule). *A rule is in the form $\{b_1 \dots b_n\} \text{ log:implies } \{h\}$. where $b_1 \dots b_n$ and h are quad patterns.*

The quad patterns $b_1 \dots b_n$ 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.

Definition 5 (Program). *A program P at context C_P is a finite set of facts and rules accessible on the web by dereferencing C_P .*

⁶ currently a W3C Working Draft

⁷ <http://www.w3.org/2000/10/swap/log#>

If the context C_P is a query-able endpoint it is possible to push selections during query processing and retrieve only the necessary information to answer a query, thus reducing the amount of data transferred.

Example 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. Figure 3 shows a rule which is stored at `exo:dblp` and relates photos in `exc:foaf` to people in `exo:dblp`, given identical homepage URIs.

```

@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix yars: <http://sw.deri.org/2004/06/yars#> .
@prefix log: <http://www.w3.org/2000/10/swap/log#> .

{{ ?x foaf:homepage ?hp .
  ?x foaf:depiction ?img . } yars:context exc:foaf .
  ?y foaf:homepage ?hp .
} log:implies {
  ?y foaf:depiction ?img .
} .

```

Fig. 3. Rule to relate photos in the FOAF data source to people in DBLP.

3.3 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 recursive rules.

Definition 6 (Immediate Consequence). Let P be a program at context C_P . A fact h is an immediate consequence of P if (i) $h \in C_P$ or (ii) $\{b_1 \dots b_n\}$ `log:implies` $\{h\}$. 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 we anticipate a large number of rules, possibly recursively referencing each other. There is no way to escape the need for recursion, since there is no central control over 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, we need recursion to be able to define the semantics of transitivity.

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]. The expressivity of our language is equivalent to (recursive) datalog under the least-model semantics.

We employ the open world assumption; results to queries are not guaranteed to be complete, since complete knowledge of a huge distributed system such as the web is somewhat illusory. Thus, our framework returns sound answers but is an incomplete procedure (i.e. all answers returned are sound but we might miss some answers due to the distributed nature of the web).

4 Vocabulary Descriptions and Ontologies

So far we have operated on a sub-schema level without taking into account formal descriptions about the data. Object-oriented features such as classes, instances, properties, and inheritance can be layered on top of RDF. Vocabulary descriptions, such as RDF Schema (RDFS) [8], or ontologies, such as OWL, the Web Ontology Language [18], are commonly used to formally describe information on the web.

4.1 Related Approaches

[11] discusses the intersection of logic programming rules and OWL called description logic programs (DLP) which is the fragment common to both paradigms. The paper also argues that language layering is a desirable feature from an architectural viewpoint.

TRIPLE's notion of parameterized context in combination with a set of rules that (partially) axiomatize the semantics of RDFS can be used to derive additional information from an RDFS specification [21].

C-OWL [7] is a proposed extension to OWL with the notion of context and bridge rules. An interesting feature of this language is that its semantics makes use of the so-called local model semantics where for each context there exists a local model and a local domain of interpretation.

4.2 Axiomatic Semantics

To be able to interpret a set of triples under the semantics of e.g. RDFS or OWL DLP we need a formalization of that semantics. For RDFS, we use a set of rules that axiomatize the RDFS semantics. The main feature of RDFS is the transitivity of `rdfs:subClassOf` and `rdfs:subPropertyOf` properties, which can be encoded using our rule language. We also use rules to encode the semantics of `rdfs:domain` and `rdfs:range` for properties.

The OWL variant that is expressible in our rule language is OWL DLP. We plan to axiomatize the OWL DLP subset using N3 rules, similar to what has been done in [15]. Once the RDFS or OWL DLP rules are added to a context, queries against the RDF graph there take into account the semantics of the respective vocabulary description or ontology.

5 Discussion and Ongoing Work

We have implemented a prototype that allows to answer conjunctive queries that span multiple contexts. We are experimenting with an RDF version of DBLP (around 1.5 GB in size) and a web crawl of RDF data (around 1 GB in size). [14] has more information on the index organization and query processing techniques used in our prototype. One immediate next step is to implement the operational semantics described in Section 3.3. We plan to investigate the use of (hybrid) top-down methods such as QRGT [22] or QSQ [2].

Ongoing work includes extending the rule language with scoped negation [20]. The idea here is that we close off the world for data sources that contain complete information about a subject; then we are able to employ a form of default negation.

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.

6 Conclusion

The aim of the thesis is to investigate methods for interoperability among autonomous RDF information spaces on the web. Our framework allows to operate with only local information, that is, creating links between sources 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 rule language N3 with context to interlink data sources. We have defined both syntax and operational fixpoint semantics of the query and rule language, and have sketched a number of questions we intend to tackle as part of the Ph.D. thesis.

Acknowledgements

I gratefully acknowledge the support of my thesis advisor Stefan Decker. This work has been partially supported by the EU IST program under the DIP project (FP6-507483) and by Science Foundation Ireland (SFI/02/CE1/I131).

References

1. K. Aberer, M. Puceva, 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. J. Bailey, F. Bry, T. Furche, and S. Schaffert. Web and semantic web query languages: A survey. In *Reasoning Web, First International Summer School*, 2005.
5. T. Berners-Lee. Notation 3 – ideas about web architecture. <http://www.w3.org/DesignIssues/Notation3.html>.
6. 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.
7. P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini, and H. Stuckenschmidt. C-OWL: Contextualizing ontologies. In *Proceedings of the 2nd International Semantic Web Conference*. Springer, 2003.
8. D. Brickley and R. V. Guha. Rdf vocabulary description language 1.0: Rdf schema. W3C Recommendation, Feb. 2004. <http://www.w3.org/TR/rdf-schema/>.
9. P. P. de Siliva and D. McGuinness. Knowledge provenance infrastructure. *IEEE Data Engineering Bulletin*, 26(4):26–32, 2003.
10. 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.
11. B. N. Grosz, I. Horrocks, R. Volz, and S. Decker. Description logic programs: Combining logic programs with description logic. In *Proceedings of the 12th International WWW Conference*. ACM Press, 2003.
12. R. V. Guha, R. McCool, and R. Fikes. Contexts for the semantic web. In *Proceedings of the 3rd International Semantic Web Conference*. Springer, Nov 2004.
13. 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.
14. 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.
15. A. Harth and S. Decker. Owl lite- reasoning with rules, 2005. WSML Working Draft.
16. 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.
17. F. Manola and E. Miller. Rdf primer. W3C Recommendation, Feb. 2004. <http://www.w3.org/TR/rdf-primer/>.
18. D. L. McGuinness and F. van Harmelen. Owl web ontology language overview. W3C Recommendation, Feb. 2004. <http://www.w3.org/TR/owl-features/>.
19. 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.
20. A. Polleres. Logic programs with contextually scoped negation. In *20th Workshop on Logic Programming (WLP 2006)*, February 2006.
21. M. Sintek and S. Decker. Triple - an rdf query, inference, and transformation language. In *Proceedings of the 1st International Semantic Web Conference*. Springer, 2002.
22. J. D. Ullman. *Principles of Database and Knowledge-Base Systems*, volume 2: The New Technologies. Computer Science Press, 1989.
23. 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.