SPOTLIGHT

Framework for the Semantic Web:

An RDF Tutorial

Stefan Decker, Prasenjit Mitra, and Sergey Melnik
Stanford University

RDF provides a data model that supports fast integration of data sources by bridging semantic differences.

The current Web supports mainly human browsing and searching of textual content. This model has become less and less adequate as the mass of available information increases. What is required instead is a model that supports integrated and uniform access to information sources and services as well as intelligent applications for information processing on the Web. Such a model will require standard mechanisms for interchanging data and handling different data semantics.

The Resource Description Framework is a step in this direction. RDF provides a data model that supports fast integration of data sources by bridging semantic differences. It is often used (and was initially designed) for representing metadata about other Web resources such as XML files. However, representing metadata[1] about the Web is not different from representing data generally. Thus, RDF can be used as a general framework for data exchange on the Web.

The RDF Data Model

The Resource Description Framework Model and Syntax Specification,[2] which became a World Wide Web Consortium (W3C) Recommendation in February 1999, defines the RDF data model and an XML-based serialization syntax. By separating the data model from the syntax needed to transport RDF data in a network, the specification allows an RDF-aware system to access a new non-RDF data source. To do this, an RDF adapter first assigns unique resource identifiers (URIs) to resources in the non-RDF data source (when URIs are not already available) and then generates statements describing resource properties. The RDF data model's use of URIs to unambiguously denote objects, and of properties to describe relationships between objects, distinguish it fundamentally from XML's tree-based data model.This simple but effective mechanism supports a general approach to representing and integrating information, as it provides the least common denominator for all information models.

Core of the Model

The RDF data model vaguely resembles an object-oriented data model. It consists of *entities*, represented by unique identifiers, and binary relationships, or *statements*, between those entities. In a graphical representation of an RDF statement, the source of the relationship is called the *subject*, the labeled arc is the *predicate* (also called property), and the relationship's destination is the *object*. Both statements and predicates are *first-class objects*, which means they can be used as the subjects or objects of other statements (see the section on reifying statements).

The RDF data model distinguishes between *resources*, which are object identifiers represented by URIs, and *literals*, which are just strings. The subject and the predicate of a statement are always resources, while the object can be a resource or a literal. In RDF diagrams, resources are always drawn as ovals, and literals are drawn as boxes.

Figure 1 shows an example statement, which can be read as: "The resource http://www.daml.org/projects/#11 has a property hasHomepage (described in http://www.semanticweb.org/schema-daml01/#hasHomepage) the value of which is the resource http://www-db.stanford.edu/OntoAgents." The three parts of this statement are the subject http://www.daml.org/projects/#11; the predicate http://www.semanticweb.org/schema-daml01/#hasHomepage; and the object http://www-db.stanford.edu/OntoAgents. A set of statements can be visualized as a graph. The graph in Figure 2 extends the statement in Figure 1 by adding the property http://purl.org/dc/elements/1.1/Creator with value Stefan Decker (a literal).

It might seem strange that predicates are also resources with URI labels, but it eliminates ambiguities that arise

 1089-7801/00/$10.00 ©2000 IEEE

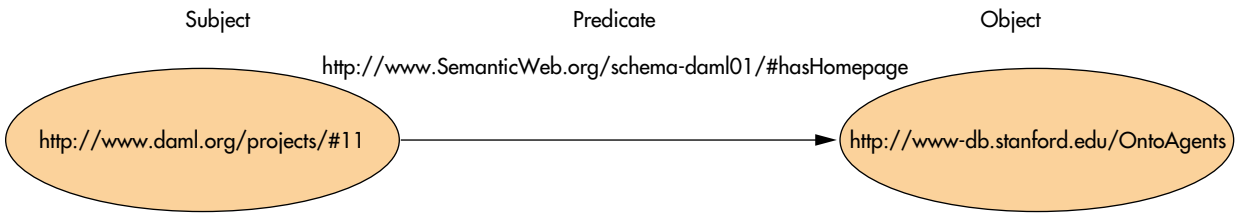

Figure 1. A simple RDF statement. The resource "http://www.daml.org/projects/#11" is a subject with a property "http://www.SemanticWeb.org/schema-daml-01/#hasHomepage." The value of the property is the object "http://www-db.stanford.edu/OntoAgents."

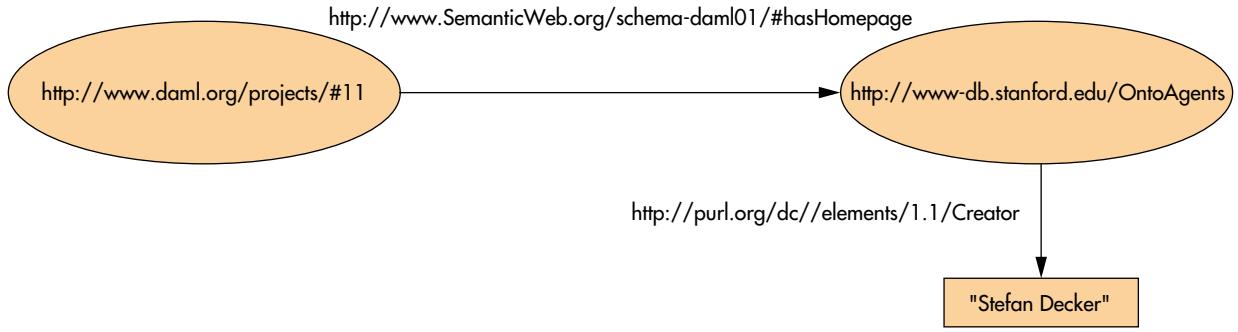

Figure 2. RDF data model graph. The property "http://purl.org/DC/#Creator" with value "Stefan Decker" (a literal) is joined with the simple statement in Figure 1 to form a graph.

from using only word identifiers. For example, vocabulary providers can define different versions of the predicate hasHomepage. URIs provide unique identifiers for each version.

Additional Vocabulary

The XML-namespace syntax[3] is used to abbreviate URIs in statements. For instance, we can define the substitution of the namespace-prefix sw for http://www.SemanticWeb.org/schema-daml01/#, and then write simply sw:hasHomepage. In this tutorial, we use the namespace prefix rdf for vocabulary defined in the RDF model and syntax specification.[2] The prefix can be expanded to http://www.w3.org/1999/02/22-rdf-syntax-ns#, which is the namespace defined for RDF-specific vocabulary.

The rdf namespace defines the property "rdf:type" to denote type relationships between two resources. The rdf:type property is particularly useful in conjunction with the Class and subClassOf vocabulary defined in the RDF Schema Specification,[4] where rdf:type is used like the instanceOf relationship in object-oriented languages.

Representing collections. We often need to make statements about collections of resources, for example, the members of a project. We could make single statements about each element of the collection; however, if we wish to make a statement about all members of a project (where the individual members might change), we must use a *container*.

RDF defines three types of containers that can represent collections of resources or literals:

- *Bags* are unordered lists, for example, a class roster where the order of student names is not important. Bags don't enforce set semantics, so a value can appear several times in a Bag.
- *Sequences* are ordered lists. Sequences represent, for example, a batch of jobs submitted to a processor, where the job order is important. Like Bags, Sequences permit duplicate values.
- *Alternatives* are lists from which the property can use only one value. We could use alternatives to express, for example, the options of flying or driving from San Francisco to San Diego, and an application could choose a suitable alternative.

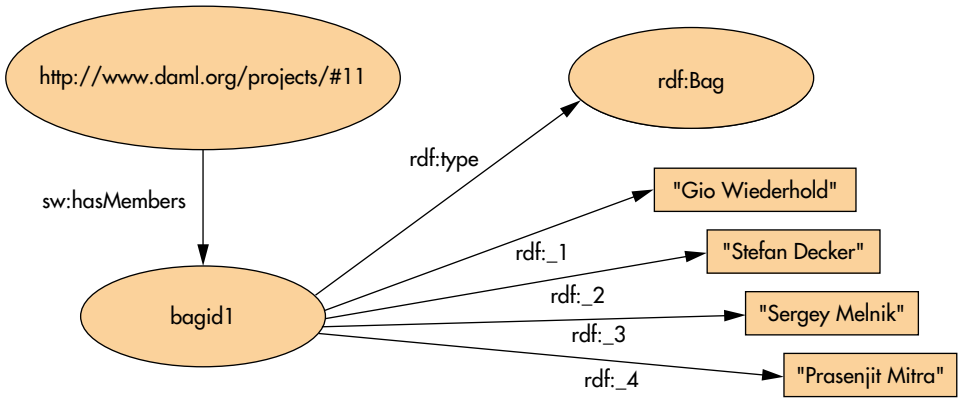

Figure 3. An rdf:Bag specifying the members of the DAML project.

In Figure 3, the resource http://www.daml.org/projects/#11 has the property sw:hasMembers, whose value is a container of the type bag. The bag is represented by an intermediate node. To refer to the intermediate node, you can give it an identifier like bagid1. An rdf:type arc specifies bagid1 as an rdf:Bag, and ordinal properties (rdf:_1 through rdf:_4) point to literals representing members' names.

Reifying statements. In RDF, we want to be able to make statements about statements. To refer to a statement, we need to treat it like a resource. The process of associating a statement and a specific resource representing the statement is formally called *reification*. The statement that has been modeled as a resource is called a *reified statement*.

Consider, for example, the following statement: "http://www.daml.org/projects/#11 is an NSF-funded project." We want to oppose this statement because it is not true (the project is funded by DARPA). To represent the statement, "The members of the project oppose the statement that http://www.daml.org/projects/#11 is an NSF-funded project," we must reify it by introducing a new resource (node in the graph) to represent the original statement. We type the new resource using the RDF-defined resource rdf:Statement, and then model the statement's subject, object, and predicate with the special properties (rdf:subject, rdf:object, and rdf:predicate) defined in the RDF specification for this task.

As shown in Figure 4, the reified statement's rdf:subject property is the URI of the OntoAgents project; the rdf:object property value is the original statement's object (the NSF identifier); and the rdf:predicate property value is the resource representing the predicate in the original statement (sw:fundedBy).

The only thing missing now is the statement about the reified statement: "All members of the OntoAgents project oppose the original statement." To model this, we introduce an additional arc, labeled sw:opposedBy, which ends at the rdf:Bag resource defined in Figure 3.

RDF Schema

The RDF Schema Specification,[4] which became a W3C candidate recommendation in March 2000, is an RDF application that introduces an object-oriented, extensible type system to RDF. RDF Schema provides means to define property domains and ranges, and class and subclass hierarchies.

In RDF Schema a property is not "local" to a class, as an attribute usually is in object-oriented languages. Instead, properties are global and described in terms of the classes they connect. To illustrate the use of RDF Schema, we will now define the vocabulary used in the previous examples. We use the namespace prefix rdfs to abbreviate the RDF Schema namespace identifier http://www.w3.org/2000/01/RDF schema#.

Figure 5 (on page 72) depicts an RDF schema that defines the class sw:Project and two properties, sw:hasHomepage and sw:hasMembers. The class is defined by typing it with the resource rdfs:Class, which is defined as a metaclass in the RDF Schema Specification. sw:Project is also defined as a subclass of rdfs:Resource, which is the most general class in the RDF Schema class hierarchy. The rdfs:subClassOf property is considered to be transitive.

All properties are defined by typing them with the rdfs:property resource class. Furthermore, a property's domain and range can be restricted using the rdfs:domain and rdfs:range properties. For exam-

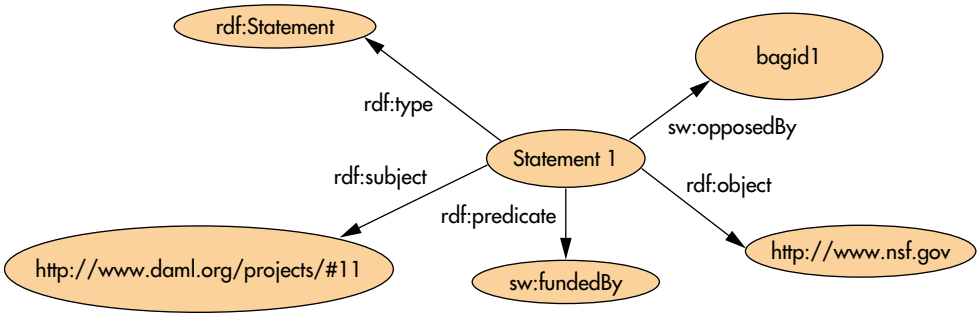

Figure 4. Graphical representation of a reified statement. In the statement, the members of the OntoAgents project oppose the statement that says the OntoAgents project is funded by NSF.

ple, the property sw:hasHomepage has the domain sw:Project and a range rdfs:Resource. Using these definitions, RDF data can be tested for compliance with a particular RDF-schema specification.

The RDF Schema Specification defines additional modeling primitives:

- rdfs:label defines a human-readable name format;
- rdfs:comment lets developers make comments;
- rdfs:subPropertyOf indicates that a property is usually subsumed by another property (for example, the property fatherOf is subsumed by the property parentOf because every father is also a parent);
- rdfs:seeAlso and rdfs:isDefinedBy indicate related Web pages that contain additional RDF information (for example, a schema);
- rdfs:ConstraintResource and rdfs:ConstraintProperty define advanced constraint mechanisms that are not covered by the RDF Schema (for example, we could define a cardinality constraint by defining a property maxCardinality, which is a subclass of rdfs:ConstraintProperty with a domain of rdf:Property and range of Integer). This is an extensibility mechanism only, which allows today's RDFS processors to determine that there are constraints they don't understand.

As a convention, application developers should put an RDF schema document declaring the vocabulary associated with a particular namespace at the namespace URI. The role of the schema document is similar to that of a contract: By delivering an RDF schema, the application developers guarantee that their application can work with RDF instance data complying with the RDF schema.

However, the language provided by RDF Schema is weaker than other modern knowledge representation languages. For example, there is no standardized way to describe cardinality constraints. Recent approaches like DAML (http://daml.semanticweb.org) and OIL (http://oil.semanticweb.org) extend RDF Schema and provide full-fledged ontology modeling languages.

RDF Syntax

The interchange of data represented in RDF must be facilitated through a concrete serialization syntax. XML is an obvious choice, and the RDF specification uses it. However, the RDF data model is not tied to a particular syntax. It can be expressed in any syntactic representation; it can also be extracted from non-RDF data sources. The XML serialization syntax of RDF is hard to understand, but RDF application programming interfaces (APIs) are supposed to shield developers from the details of the serialization syntax and to let them handle RDF data as graphs (see the sidebar, "RDF Tool Support and Deployments" on page 73). In this tutorial, we will illustrate the syntax with examples, but we will not cover all possible uses of the XML serialization syntax defined in the RDF specification.

The specification suggests two syntaxes for serializing RDF data in XML: *abbreviated* and *standard*. Both use the XML namespace mechanisms to abbreviate URIs. The RDF syntax is defined to look like existing XML documents. For example, it tries to be "human-readable" and closely resembles data representation in the simple object access protocol (SOAP).[5] Figure 6 shows the XML serialization of the RDF graphs from Figures 2. (Note that we have left out the namespace declarations.)

RDF can be embedded in an XML document. To make the segment in Figure 6 a correct RDF document, we just have to add the namespace declaration shown in Figure 7. RDF code usually starts and

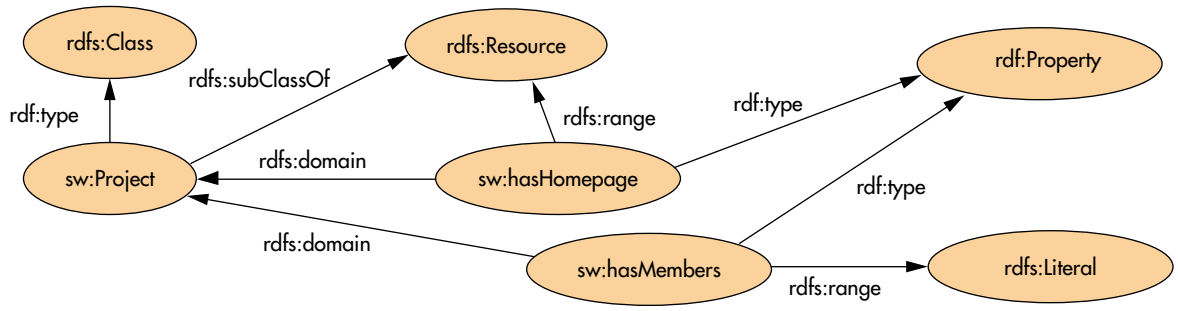

Figure 5. RDF schema definitions defining the vocabulary used in the previous figures.

```
<Project rdf :about="http://www.daml.org/projects/#11">
    <hasHomepage>
        <rdfs:Resource rdf:ID="http://www-db.stanford.edu/OntoAgents">
            <dc:Creator>Stefan Decker</dc:Creator>
        <rdfs:Resource>
    </hasHomepage>
</Project>
```

Figure 6. Abbreviated XML serialization defining an instance of "Project," which has a home page. The home page itself has a creator. Please note that namespace declarations are omitted.

```
<?xml version='1.0'?>
<rdf:RDF
    xmlns="http://www.SemanticWeb.org/schema-daml01#"
    xmlns:rdf= "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:dc="http://purl.org/ dc/elements/1.1/">
    …
</rdf:RDF>
```

Figure 7. RDF namespace declaration. Adding the RDF declaration to the XML serialization in Figure 6 forms a correct RDF document.

ends with indicated <rdf:RDF> and </rdf:RDF> tags, but the tags are optional if the processing application already knows to expect RDF data.

An RDF description starts either with a typing identifier (such as project, to denote the type of the resource described) or simply with rdf:Description, if no explicit typing is given. Then an ID can be defined, usually with the rdf:about or rdf:ID XML attribute, to enable references to the defined resource.

The next level of nested tags (for example, hasHomepage in Figure 6) gives properties of the resource denoted by the ID. The values of the properties are RDF resources, which can again have other properties (for example, the resource http://www.db.stanford.edu/OntoAgents in Figure 6 has a property dc:Creator with a certain value).

Conclusion

With its built-in notion of resources and relationship between resources, RDF aims to fulfill the promise to populate the Web with machine-processable information. The simplicity of the RDF data model makes representing data straightforward, and more sophisticated representation languages like the Unified Modeling Language (http://www.omg.org/technology/uml/ and http://wwwdb.stanford.edu/~melnik/rdf/uml/) or Description Logics[6] can be defined on top of RDF.

In a sense, RDF is the lowest common denominator for establishing interoperation between Web applications. Being object-oriented, it has a more suitable data model for exchanging information than XML, and it is extremely flexible for defining new vocabularies. RDF is the ideal tool for the next phase in the development of the Web, when vocabularies and vocabulary marketplaces will become more important.

Further information about these topics can be found at http://www.w3.org/rdf/ and http://www.semanticweb.org. ■

Acknowledgments

We thank Ora Lassila for detailed comments on a draft of this tutorial.

References

1. O. Lassila, "Web Metadata: A Matter of Semantics," *IEEE Internet Computing*, vol. 2, no. 4, July/Aug. 1998, pp. 30-37.
2. O. Lassila and R. Swick, "Resource Description Framework (RDF) Model and Syntax Specification," World Wide Web Consortium Recommendation, Feb. 1999; available at http://www.w3.org/TR/REC-rdf-syntax/.

3. T. Bray, D. Hollander, and A. Layman, "Namespaces in XML," W3C Recommendation, Jan. 1999; available at http://www.w3.org/TR/REC-xml-names/.
4. D. Brickley and R. Guha, "Resource Description Framework (RDF) Schema Specification," W3C Candidate Recommendation, Mar. 2000; available at http://www.w3.org/TR/2000/CR-RDF-schema-20000327.
5. D. Box et al., "Simple Object Access Protocol (SOAP) 1.1," W3C note, May 2000; available at http://www.w3.org/TR/2000/NOTE-SOAP-20000508.
6. I. Horrocks et al., "The Ontology Interchange Language (OIL)" tech. report, Free University of Amsterdam, 2000; available at http://www.ontoknowledge.org/oil/.

Stefan Decker is a postdoctoral fellow in the computer science department at Stanford University. His research interests include intelligent information integration, knowledge representation, and inferencing on the Web. He consults frequently on RDF, XML, and interoperability issues.

Prasenjit Mitra is a doctoral candidate at Stanford University. He obtained a bachelor of technology with honors in computer science and engineering from the Indian Institute of Technology, Kharagpur, and an MS in computer science from the University of Texas at Austin. His research interests are the semantics of knowledge, knowledge bases, and databases and logic.

Sergey Melnik is a visiting researcher in the computer science department at Stanford University. His research interests include knowledge representation and database systems for the Web, information integration, and digital libraries.

Readers can contact the authors at {stefan,prasen9,melnik}@db.stanford.edu.

RDF Tool Support and Deployments

Various tools are already available for handling RDF and RDF schemas. We list just a few here; a more complete list is available at the W3C's RDF home page at http://www.w3.org/RDF/.

The Protégé-2000 Ontology Editor supports the creation of RDF schemas and RDF data. It is a system for knowledge-base design and knowledge acquisition using Java. Protégé-2000 was developed at Stanford University and is available under the open-source Mozilla public license at http://www.smi.stanford.edu/projects/protege/.

Sergey Melnik has implemented a Java-based RDF API that includes an XML and RDF parser, schema-based validation facilities, cryptographic digests of RDF models and statements, UML support on top of RDF, and support for RDF schema handling in Java. The API is available under an open-source license from http://www-db.stanford.edu/~melnik/rdf/api.html.

SiLRI is a lightweight deductive database that can reason using RDF metadata. It is also Java based and available under an open-source license from http://www.aifb.uni-karlsruhe.de/~sde/rdf.

Ramanathan V. Guha implemented an RDF Database (RDFDB), based on the Berkeley DB. It supports a graph-oriented API using a textual query language similar to SQL, and aims to scale to millions of nodes and triples. RDFDB can be downloaded from http://rdfdb.sourceforge.net.

The FRODO RDFSViz provides a visualization service for ontologies represented in RDF Schema. It uses the Java RDF API implementation from Sergey Melnik and the Graphviz graph drawing program (AT&T and Lucent Bell Labs). The FRODO RDFSViz tool is available at http://www.dfki.uni-kl.de/frodo/RDFSViz/.

Current deployments of RDF technology include:

- The Platform for Internet Content Selection (PICS) is a system for associating metadata (PICS "labels") with Internet content. A specification for describing PICS metadata in RDF is available at http://www.w3.org/TR/rdf-pics.
- Netscape's Open Directory Project, available as an RDF dump at http://dmoz.org/rdf/.
- The open.gov.uk service, a first entry point to UK public sector information on the Internet, uses the Dublin Core RDF vocabulary to describe each of the resources available on the site. The service is available at http://open.gov.uk.
- The California Environmental Resources Evaluation System (CERES) Thesaurus Effort and U.S. Geological Survey Biological Resource Division, which uses RDF to build digital thesauri, is available at http://ceres.ca.gov/thesaurus/.
- LastMileServices (http://www.lastmileservices.com), a telecommunications startup, is building an RDF-based vocabulary for the telecommunications sector.
- The RDF Schema, available at http://www.cim-logic.com/cim-rdf/CIM-schema-cimu09a.rdfs, is currently in the process of becoming an International Electrotechnical Commission standard. The RDF Schema format is the basis for an OMG standard created within the Utility Domain Task Force.