Resource Description Framework: Metadata and Its Applications

K. Selçuk Candan, Huan Liu, and Reshma Suvarna
Department of Computer Science & Engineering
Arizona State University
Tempe, AZ 85287-5406
{candan,hliu,reshma.suvarna}@asu.edu

ABSTRACT

Universality, the property of the Web that makes it the largest data and information source in the world, is also the property behind the lack of a uniform organization scheme that would allow easy access to data and information. A semantic web, wherein different applications and Web sites can exchange information and hence exploit Web data and information to their full potential, requires the information about Web resources to be represented in a detailed and structured manner. Resource Description Framework (RDF), an effort in this direction supported by the World Wide Web Consortium, provides a means for the description of metadata which is a necessity for the next generation of interoperable Web applications. The success of RDF and the semantic web will depend on (1) the development of applications that prove the applicability of the concept, (2) the availability of application interfaces which enable the development of such applications, and (3) databases and inference systems that exploit RDF to identify and locate most relevant Web resources. In addition, many practical issues, such as security, ease of use, and compatibility, will be crucial in the success of RDF. This survey aims at providing a glimpse at the past, present, and future of this upcoming technology and highlights why we believe that the next generation of the Web will be more organized, informative, searchable, accessible, and, *most importantly*, useful. It is expected that knowledge discovery and data mining can benefit from RDF and the Semantic Web.

Keywords

Resource Description Framework (RDF), Metadata, XML, Web, Semantic Web

1. INTRODUCTION

Although the World Wide Web has revolutionized the means of data availability, with its current structure, it falls short of being a reliable and efficient tool for global information access. While users can access an unprecedented amount of data on the Web, they increasingly find it difficult to retrieve relevant information. Search tools on the Web provide indexes to help users. However, sorting through ever-increasing data and identifying what may be relevant for a particular user is not trivial. In order to make this task easier , current search engine technologies work mostly at the page-level: given a query, a search engine returns pages each of which *alone* answers the query. Unfortunately, the returned list of pages may miss some relevant information or may contain many irrelevant pages. Consequently, the user has to manually filter the result list and/or change the query to re-search for relevant pages. Therefore, the Web's current information retrieval model makes it extremely difficult for users to find and use relevant information.

Universality, the property of the Web that makes it the largest data and information source in the world, is also the property behind these challenges. Data available on the Web covers diverse structures, formats, as well as content. However, the Web lacks a uniform organization scheme that would allow easy access to data and information. Clearly, a framework that would help search engines and other information access and integration tools to organize the data available on the Web would be as revolutionary as the creation of the Web itself. Such a framework would render the Web manageable, retrievable, and uniform, while not taking anything away from its universality. Hence, such a framework can also facilitate tasks, such as data mining [20; 13] and web mining [54; 32].

To make information access easier, such a framework would

need to state, explicitly, what a Web page (or any other data resource on the Web) actually contains. In other words, such a framework would need to be based on *metadata* (data about data) that describes content of Web resources. In fact, keyword indexes used by search engines are nothing but simple data structures for metadata that describe the textual content of pages. Since tools and mechanisms that will allow page creators to describe the semantic metadata are not widely available, search engines have to rely on the keyword content or human experts to index pages. Although some keyword-based classification, such as latent semantic indexing [18; 3], and link structure based ranking techniques, such as PageRank [46; 11], are used to increase the quality of the results, search engines are far from understanding and using actual semantic content of web resources for answering user requests. Since the heuristics used in the search process are not perfect, search engines and other information access tools cannot provide highly efficient access to information on the Web.

If authors could easily associate metadata with each Web resource (that can be represented with a uniform resource identifier or URI) they create, then this metadata could be used by information access and integration engines to increase their efficiency and precision. In order to enable this, the metadata format used by different authors must be compatible with each other. In addition, in order to enable the development of future applications with different data and information needs, the format must be generic and the metadata schemas must be extensible.

Substantial efforts have been taken in this direction, and a number of frameworks have evolved over the years. The most promising of these is the Resource Description Framework (RDF). RDF provides application developers with a solid foundation for the description of metadata for the next generation of interoperable Web applications. The RDF Model and Syntax Specification (REC-rdf-syntax) is currently a W3C[1] recommendation [35], and the RDF Schema Specification (PR-rdf-schema) is a proposed recommendation [10].

The long-term goal of RDF is to link different applications and Web resources into a new global network, the Semantic Web [6]. The Semantic Web is envisioned to be the next generation of the current Web, wherein the information about Web resources is represented in a detailed and structured manner, using RDF. Ontologies[2], which describe the context in which metadata is applied, are used to link, compare, and differentiate information provided by various Web resources. Once all Web resources are described in a uniform way, the information exchange between individuals and applications will be possible to a much greater extent [6].

This paper is a survey of current RDF status. The paper motivates the need for metadata of data on the Web in Section 1. Section 2 throws light on the background of RDF. Section 3 illustrates the RDF model and RDF components. Section 4 reviews some available RDF Tools for metadata generation, Application Program Interfaces and RDF databases. Section 5 discusses RDF real-world applications and its relationship to the Semantic Web. Section 6 concludes the paper with future work and challenges.

2. BACKGROUND OF RDF

At the present stage of the Web's evolution, most traffic on the Internet is between human consumers using Web browsers and content providers using Web servers. This means that to find and retrieve data, human intervention is often required. As businesses move more of their daily operations online, computer-to-computer peer services are growing [24]. Therefore the need is increasing for developing a model which will bring structures to descriptions of the Web content, thus creating an environment where the tasks such as searching the Web could be automated.

Many proposals were made to the World Wide Web Consortium (W3C) for representation of Web-related metadata. Initial solutions were based on the `<META>` tag of the HTML. Later on, with the advent of XML[3], more descriptive content tags have been introduced. Currently, many companies, such as Microsoft, IBM, Motorola, Netscape, Nokia, OCLC, are actively participating in the field of metadata framework developments. Here is a brief look at the development of metadata frameworks.

The first solution to be developed was naturally the HTML metadata tag, `<META>`, which resides within the `<HEAD>` element. `<META>` has two attributes, `NAME` and `CONTENT`, which can be used to store the metadata schema. Due to its

[1] The World Wide Web Consortium (W3C) is an international consortium of companies involved with the Internet and the Web. The W3C was founded in 1994 by Tim Berners-Lee, the original architect of the WWW. Its purpose is to develop open standards so that the Web evolves in a single direction rather than being splintered among competing factions [38].

[2] An ontology is a document or file that formally defines the relations among terms. The most typical kind of ontology for the Web has a taxonomy and a set of inference rules.

[3] Extensible Markup Language, XML, is a W3C-endorsed standard for document markup [24].

very simplicity, this solution cannot be used for describing complex properties.

The advent of XML, the Extensible Markup Language, enabled a stronger functional scheme. XML defines a generic syntax used to mark up data with simple, human readable tags. It provides a standard format for computer documents. XML tags can be used for describing arbitrary properties of Web resources. Although this provides flexibility and expressive power to the metadata description framework, it also complicates the handling of metadata as different frameworks could have different sets of rules and properties to represent data. In fact, the resulting incompatibility turned out a major hurdle in XML-based frameworks. Namespace was proposed to overcome the problem.

Companies like Microsoft and Netscape were actively involved in the metadata framework development. Channel Definition Framework (CDF) is the industry's first channel framework on the Web and is Microsoft's major contribution to the metadata initiative. It introduces and describes channels at high level using HTML for primary content description. Each `<CHANNEL>` is composed of multiple `<ITEM>` elements, which describe HTML pages.

In 1997, along the line of the Web collections idea, Netscape submitted a new proposal, titled "Meta Content Framework", to W3C [23]. The two principles on which the meta content framework (MCF) is based are as follows:

- There is no distinction between the representation needs of data and metadata, and
- For interoperability and efficiency, schemas for different applications should share as much as possible in the form of data structure, syntax, and vocabulary.

MCF is based on a system of objects, property types, and properties. This framework offers a common data model and vocabulary, making it possible to query and manage metadata, without having to fully understand the semantics or vocabulary behind it.

The culmination of all these various frameworks was the creation of the RDF in 1997 [34]. RDF has drawn influence from several different sources. The main influences have come from the Web standardization community itself in the form of HTML metadata, the library community, the structured document community in the form of SGML (Standard Generalized Markup Language) and more importantly XML, and also the knowledge representation community. Other areas of technology also contributed to the RDF design such as object-oriented programming and modeling languages, as well as databases. RDF is still evolving.

Note, finally, that the first principle of MCF, described earlier, raises an important question: "If there is no distinction between the representation needs of data and metadata and if we are using XML to represent data, why do we need RDF?" The answer to this question is hidden in the meaning of "representation". As described above, XML simply is a markup language for formatting textual documents in a human readable format. However, the hierarchical structure provided by XML as well as the fact that arbitrary references are allowed within XML allowed it to be used to capture certain types of data models as well. However, RDF provides a richer data model where entities and relationships can be described. Unlike traditional object-oriented data models and XML, the relationships in RDF framework are first class objects, which means that relationships between objects may be arbitrarily created and be stored separately from the two objects. This nature of RDF is very suitable for dynamically changing, distributed, shared nature of the Web. In other words, RDF provides a more suitable data model than XML and its rich modeling tools, such as XML Schemas [55]. Also, there is more to RDF than the underlying modeling tool: It is designed to provide a framework that ensures interoperability between metadata frameworks. Hence, it allows applications to exchange machine understandable information on the Web, promoting unilateral organization of Web resources by their suppliers.

In the next section, we will focus on the modeling capabilities of RDF.

3. RDF MODEL AND RDF COMPONENTS

The Resource Description Framework (RDF) is an XML-based language [52] for describing information contained in a Web resource. A resource can be a Web page, an entire Web site, or any item on the Web that contains information in some form [25]. RDF enables the encoding, exchange, and reuse of structured metadata [43]. It allows for metadata interoperability through the design of mechanisms that support common conventions of semantics, syntax, and structure. RDF makes no assumption about a particular application domain, nor defines the semantics of any particular application domain. The definition of the mechanism is domain neutral, yet the mechanism is suitable for describing information about any domain [35]. RDF can be used in a variety of application ar-

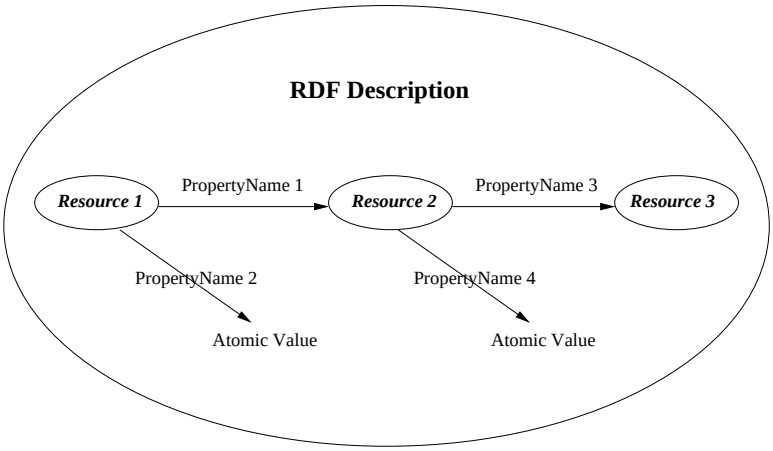


Figure 1: Overview of the RDF model.

eas including [27]:

- Resource Discovery - RDF will enable search engines to more easily discover resources on the Web.
- Cataloging - RDF will enable users to better describe the content and content relationships available at a particular Web site, page, or digital library.
- Intelligent Software Agents - RDF will facilitate knowledge sharing and exchange, and allow software agents to more intelligently find, filter and merge data.
- Content Rating - RDF will allow content to be rated.
- Intellectual Property Rights - RDF will allow users to more easily express and enforce intellectual property rights of Web sites.
- Privacy Preferences and Privacy Policies - RDF will allow users and Web sites to express privacy preferences and site-wide privacy policies that can be interpreted by applications.
- Digital Signatures - RDF will be a key to building the "Web of Trust" for e-commerce, collaboration, and other applications.

An RDF model consists of schemas, components, statements, containers, statements about RDF statements, as well as XML namespaces. We illustrate these elements of an RDF model in the following.

RDF Schemas

To facilitate the definition of metadata, RDF has a class system much like many object-oriented programming and modeling systems. A collection of classes is called a schema. Through shareability of schemas, RDF supports reusability of metadata definitions. Due to RDF's incremental extensibility, agents[4] processing metadata will be *able to trace the origin of schemas they were unfamiliar with back to known schemas and perform meaningful actions on metadata they were not originally designed to process*. The shareability and extensibility of RDF also allows metadata authors to use multiple inheritance to *mix* definitions and provide multiple views to their data. In addition, RDF allows creation of instance data based on multiple schemas from multiple sources. Each RDF application will use a schema to restrict its use of RDF to a deliberately limited language [35].

RDF Components

RDF is a syntax independent model for representing resources and their corresponding descriptions [34; 43]. It provides a model for describing Web resources; i.e., objects that are uniquely identifiable by uniform resource identifiers (URIs). The resources are described using property names, which express the relationships of values associated with resources [25]. Values may be atomic or may be other resources, which in turn may have their own properties. A collection of these prop-

[4] Agents are software programs that perform some information-gathering or -processing task in the background. Typically, an agent is given a small and well-defined task. Agents have become more prominent with the recent growth of the Web.

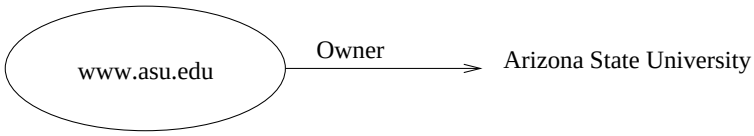


Figure 2: Visual representation of Statement 1.

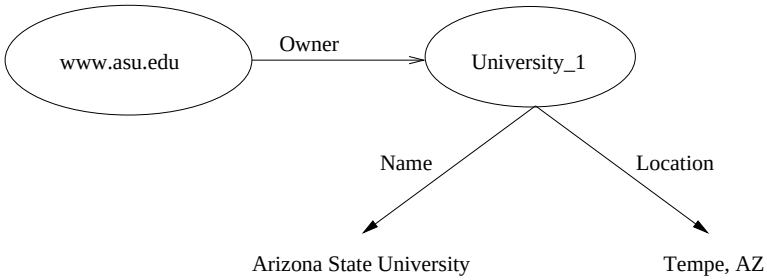


Figure 3: Alternative description of Statement 1 using more detailed metadata

erties that refer to the same resource is called a description (Figure 1) [43]. Therefore, the RDF model consists of three major *components* [35]:

- **Resources:** All things being described by RDF expressions are called resources. A resource may be an entire Web page, part of a Web page, an entire Web site, or an object that is not directly accessible via the Web page (e.g., a printed book).
- **Properties:** A property is a specific aspect, characteristic, attribute, or relation used to describe a resource. Each property has a specific meaning, defines its permitted values, the types of resources it can describe, and its relationship with other properties [10].
- **Statements:** A specific resource together with a property plus the value of that property for that resource is an RDF statement. These three individual parts of a statement are called the subject, predicate, and object, of the statement, respectively. More concrete details can be found in the following example.

Let us consider the page http://www.asu.edu (home page of the Arizona State University - ASU) as an example. We can see that this resource can be described using various *page related* content-based metadata, such as `title of the page` and `keywords in the page`, as well as *ASU related* semantic metadata, such as the `president of ASU` and its `campuses`. If we consider another related resource, http://www.eas.asu.edu/, (College of Engineering and Applied Sciences Web page at ASU), we can see that although it also can be described using similar metadata, there are some differences. For instance, this resource has additional properties, such as `a dean` and `a list of courses`. On the other hand, it lacks the information about `campuses`.

RDF Statements

Let us consider the following statement about the Web resource http://www.asu.edu/ and see how we would use RDF to describe it).

Statement 1. *"The owner of the Web site http://www.asu.edu is Arizona State University".*

Figure 2 shows how we can use RDF to express this statement using (1) a resource or *subject* (http://www.asu.edu), (2) a property name or *predicate* (owner), and (3) an atomic value or *object* (Arizona State University).

If we know more about the Arizona State University that we would like to include in our description, we could replace the atomic value (Arizona State University) by a resource (University_1) which can be further described using appropriate property names and values as shown in Figure 3.

Note in this example that some metadata (such as property names) used to describe resources are generally application

```
1  <?xml version = "1.0"?>
2  <rdf:RDF
3    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
4    xmlns:my="http://mymetadata.org/schema/">
5    <rdf:Description about="http://www.asu.edu/namespace/">
6      <my:Title>NamespaceFAQ</my:Title>
7      <my:Description>
8        This is the page of FAQ for ASU namespace.
9      </my:Description>
10     <my:Date>2001-06-14T09:46</my:Date>
11   </rdf:Description>
12 </rdf:RDF>
```

Figure 4: An RDF model for a document.

dependent, and must be associated with RDF schemas. This, however, can cause difficulties when RDF descriptions need to be shared across application domains. For example, the property *location* can be defined in some other application domain as *address*. Although, the semantics of both property names are the same, syntactically they are different. On the other extreme, a property name may denote different things in different application domains. In general, a property name may have a broader or narrower meaning depending upon the needs of particular application domains. In order to prevent such conflicts and ambiguities, the terminology used by each application domain must be clearly identified. This can be achieved by using standard metadata, such as the Dublin Core[5] [19], or by using *namespaces*.

Namespaces in RDF

RDF uniquely identifies property names by using *the XML namespace* mechanism [43]. A namespace can be thought of as a context or a setting that gives a specific meaning to what might otherwise be a general term [27]. The XML namespace provides a method for unambiguously identifying the semantics and conventions governing the particular use of property names by uniquely identifying the governing authority of the vocabulary. As humans, we do this type of namespace or context mapping quite automatically. Thus, using namespaces, RDF provides ability to define and exchange semantics among communities. With the above knowledge, we provide, in Figure 4, an example of some RDF description of a document.

[5]The Dublin Core, http://purl.org/dc/, is a standard set of ten information items with specified semantics that reflect the sort of data likely to be found in a card catalog or annotated bibliography.

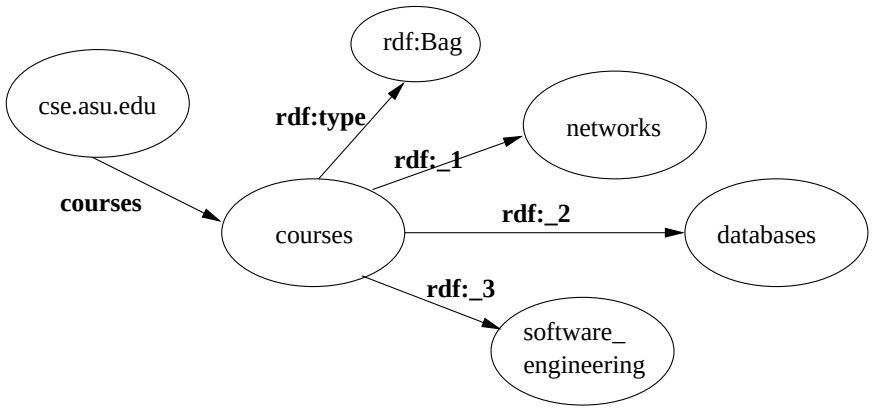
In Figure 4, Line 1 specifies the version of XML to which the document conforms. Lines 2-4 define the root element `rdf:RDF`. Here, the two namespace prefixes `rdf` and `my` are declared. These namespaces are applicable to the RDF description in lines 5-11. The URIs associated with the namespace declarations reference the corresponding schemas. Line 5 uses element `rdf:Description` (the element "Description" in the context of the `rdf` namespace) to describe the resource - the corresponding URI: `http://www.asu.edu/namespace/` - specified in attribute `about`. Line 6 uses element `my:Title` (the element "Title" in the context of the `my` namespace) to mark up a property with name `my:Title` and value "NamespaceFAQ". Lines 7-10 use other metadata elements to provide more information about the resource.

RDF Containers

Frequently it is necessary to refer to a collection of resources: for example, to list of courses taught in the Computer Science Department,or to state that a paper is written by several authors. To represent such groups, RDF provides *containers* [35] to hold lists of resources or literals. RDF defines three types of container objects to facilitate different groupings:

- **Bag** is an unordered list of resources or literals.
- **Sequence** is an ordered list of resources or literals.
- **Alternative** is a list of resources or literals that represent alternatives for the (single) value of a property.

Statement 2. "The CSE department (cse.asu.edu) offers courses: Networks, Databases, Software Engineering".




```
<rdf:RDF>
  <rdf:Description about="http://cse.asu.edu/courses">
    <my:courses>
      <rdf:Bag>
        <rdf:li resource="http://cse.asu.edu/courses/networks">
        <rdf:li resource="http://cse.asu.edu/courses/databases">
        <rdf:li resource=
              "http://cse.asu.edu/courses/software_engineering">
      </rdf:Bag>
    <my:courses>
  </rdf:Description>
</rdf:RDF>
```

Figure 5: A simple Bag container example: its graph model and RDF/XML implementation.

To represent this statement, we can use a bag description as shown in Figure 5.

Statements about RDF Statements

In addition to making statements about a Web resource, RDF can also be used for *making statements about other RDF statements*:

> **Statement 3.** "According to the ASU catalog, the CSE department offers courses: Networks, Databases, Software Engineering". It is a statement about the statement "The CSE department offers courses: Networks, Databases, Software Engineering".

In this statement, we say nothing about the courses offered in the CSE department; instead, we express a fact stated in the ASU catalog. In order to express this fact using RDF, one has to model the original statement as a resource with four properties: *subject*, *predicate*, *object*, plus *type* whose value describes the type of the new resource. For instance, in order to model the example statement, we need to attach another property (e.g., "attributedTo") with an appropriate value (here, "ASU catalog"). The corresponding RDF graph is shown in Figure 6. In effect, these higher order statements treat RDF statements as uniquely identifiable resources. This process is called reification [35] and the statement is called a reified statement. All reified statements are instances of `RDF:Statement`.

Other RDF Constructs

The RDF model does not need a special construct for descriptions since descriptions really are collections of statements. A Bag container is used to indicate that a set of statements came from the same description, thus reifying each statement of the particular description and making each reified statement a member of the Bag container [35]. RDF model intrinsically supports binary relations (a statement specifies a relation between two Web resources). Higher arity relations have to be represented using binary relations [35]. Intuitively, tools are

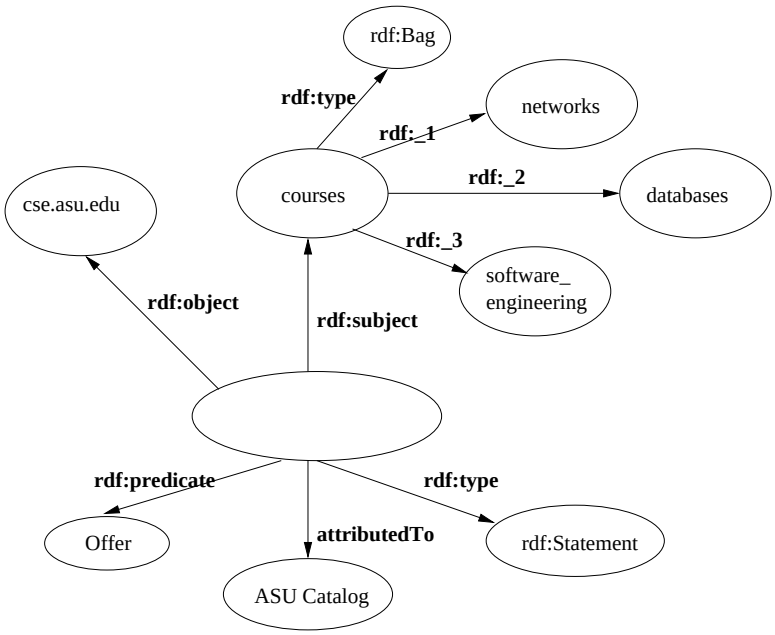


Figure 6: A statement about a statement.

necessary to extract metadata from a given Web source. Examples of RDF tools are discussed in Section 4.

4. RDF TOOLS

RDF is an evolving technology for developing information rich applications [27]. Its wide-spread adoption by the Web community, on the other hand, depends both on the expressive power it will provide for description of metadata and the availability of tools that will make RDF an easy-to-use framework. Such tools range from those that create and store metadata in RDF format to graphical user interfaces for editing RDF models. In this section, we review some existing RDF tools under three categories: *automated metadata generation tools, application program interfaces*, and *RDF databases.*

Automated Metadata Generation Tools

Metadata generation tools extract metadata from Web resources and store it in RDF for later use. Here, we provide some examples of metadata generation tools [49].

Reggie [50] is a tool capable of extracting metadata from given Web sources (Web pages). The user can select any existing schema file or can create his/her own schema files. Reggie extracts the META tags from a given URL and attempts to add them to the most appropriate fields of the chosen schema. It also allows users to create their own metadata schema files. Once the user decides the schema file to be used to retrieve the metadata, Reggie reads in the details of all the elements, their characteristics, and descriptions. Note, however, that Reggie is not limited to describing Web resources only. In fact, it can handle various forms of metadata, including Dublin Core.

DC-DOT [16] is another metadata extract tool. In contrast to Reggie where the user provides the schema file, DC-DOT specifically uses the Dublin Core schema, a metadata element set for description of electronic resources, to extract metadata from a given Web resource. DC-DOT uses the information contained in the META tags of a Web source to generate the RDF model.

Like DC-DOT, an automatic classifier is described in [29] which identifies document related metadata, such as document title, keywords, abstract, and word count. The resulting metadata is represented using an RDF schema specifically created for this purpose. Given a Web resource, an appropriate metadata element set, Wolverhampton Core, is identified based on the keywords extracted from the resource. An appropriate RDF data model and a schema are proposed in [29] to represent the element set commonly used for retrieving documents.

Note that there are only a handful of automatic metadata ex-

traction tools, and most of the available ones are fine-tuned for metadata extraction using well defined schemas, such as the Dublin Core schema. The lack of such tools points to the difficulty of understanding the semantics of web resources and automating the resource description process. Standard metadata schemas, such as Dublin Core, alleviate this difficulty considerable. However, generic metadata extraction tools, such as Reggie, which allows extraction of metadata based on user-specified schemas, are essential for scaling the use of metadata to multiple applications, each with its own needs.

Application Program Interfaces

For RDF to achieve widespread acceptance, it is necessary for application developers to have an easy-to-use interface to the RDF models [15; 41]. Application program interfaces (APIs) can help users to edit, update, and create new RDF models. *RADIX* [15] is a proposal to W3C that contains a collection of requirements an API for RDF should fulfill. Some important requirements in this proposal are that an RDF API should (1) be independent of different means of data storage (that is, it should support diverse storage mechanisms); (2) support different front-end parsers; (3) provide easy manipulation of the nodes and arcs in an RDF graph; (4) support queries about the nodes and arcs; and (5) support and provide metadata for RDF models. There are a few APIs that support RDF. The degree of adoption of the above requirements change from API to API.

Redland [4] is a library that provides a high level object-oriented interface for RDF. Redland implements each RDF concept in its own class. The modular, object-oriented nature of Redland enables the end-user to plug-in different parsers and storage mechanisms as suitable. Redland provides interfaces for the C language. *Jena* [28], on the other hand, is a java API for RDF being developed by HP. It supports both statement- and resource-centric views of RDF. Hence, it is possible to treat RDF models both as sets of triples and as sets of resources/properties, respectively. Jena also supports multiple implementations.

The Generic Interoperability Framework, GINF [39], uses RDF as a generic representation for protocols, languages, data, and interfaces. It uses an RDF interface which not only allows creation and manipulation of RDF models, but also access to these models through a SQL like query interface. To certain degree, this interface formed the basis of the W3C RDF Interest Group's RDF API proposal to the W3C RDF Interest Group [40].

Other RDF APIs include CARA [31] developed as part of the CARMEN project and Mozilla API [48], developed as part of the Mozilla open-source web browser. More recently, another open-source framework, RDF.NET, capable of parsing and processing RDF models for Microsoft's .NET platform has been proposed [47].

The multitude of APIs for RDF points to the need and the interest in the community for frameworks capable of manipulating RDF models. Although the W3C standardization activities and emergence of efforts, like RDF.NET, which aim to provide comprehensive APIs, initiated discussions on converging different API implementations into a uniform framework, the largely varying needs of the end-users do not lend themselves to such a uniform API.

RDF Databases

One of the most important functionalities of an RDF API is to enable access to RDF models and the relevant Web data. Consequently, many of the APIs described above, such as Redland, provide mechanisms for storage and retrieval of RDF data. This brings forth the necessity of having an appropriate database technologies as well as easy and efficient query and inference mechanisms. Below, we discuss some developments in this direction.

As we have discussed earlier, RDF provides a rich data model, capable of describing both web entities and the relationships between them as first class objects. Therefore, a storage and retrieval system for RDF has to address two major challenges: (1) developing appropriate query mechanisms for RDF models, and (2) merging together *distributed*, independently-created, dynamic RDF models.

rdfDB [22] aims providing a solution to the first challenge: it aims to be a database system capable of supporting a graph oriented API via a textual query language. It aims to be scalable to millions of nodes and triples, to support RDF schemas, and to provide basic forms of inference. The query language used by rdfDB is similar to SQL; hence most existing database users can adopt rdfDB easily.

In rdfDB, a collection of RDF triples forms a database. The basic database operations remain essentially the same, but are modified to accommodate the RDF model. The normal database commands for insertion, deletion, and querying are done in rdfDB in the following ways:

- To create a new database called myrdf, the "Create database `myrdf`" command is used.
- The table manipulation operations, like insertion and deletion of entries, are similar to those of traditional

databases. For example,

- "Insert into `myrdf` (author, http://myrdf.org/, John)" inserts a triple into the database `myrdf`.
- "Delete from `myrdf` (author, http://myrdf.org, John)" deletes a triple from the database.

As discussed earlier, RDF vocabularies may come from different namespaces. rdfDB creates URIs by concatenating the namespace URI with "#" and the element name. The query commands are also similar to those in SQL. For example, consider the `myrdf` database described above. If the user wants to retrieve the resource where the author is "John". The query is written using the following SQL-like statement: "Select ?p from `myrdf` where (author ?p John)".

However, without a richer inference mechanism, rdfDB does not extend beyond being a simple SQL-based query language that can be implemented on top of a simple relational data model. In fact, RDF querying and storage mechanisms are not necessarily tied to each other: Irrespective of the query language used for accessing RDF data, RDF models can be stored using other more established data models and mechanisms. For instance, Inkling [44] translates RDF queries described in SquishQL [45] into SQL queries that can be executed on any relational database. Various other systems also built RDF query engines on top of relational databases. The actual strategy to be used to map the RDF triples onto tables in a relational database varies depending on the underlying implementation [42].

Although using relational approaches provides a simple solution to storage and retrieval of RDF models, the second challenge (i.e., merging together independently-created RDF data that provides a unified model enabling us to identify relevant web objects) can be explored only if we have a query mechanism that can draw further inference from the available metadata [17; 5]. Algae [1] query language, for instance, is built on top of the Algernon, a rule compiler. Rich, rule-based systems can provide better access to full potential of the rich RDF model. Because of reification, RDF components have implied properties. Hence while designing a query system or query language for RDF, we have to consider the RDF model complexities and also the implicit representation. We may also want the query system to be capable of querying not only RDF data but also the RDF schemas. As a result, the query issue becomes complex and the query language to be used should be able to tackle complex semantic queries. Thus the traditional query techniques are not sufficient for full fledged use of RDF. Logical inference service [53; 37] is a necessary part of the query services to be provided. For example given an RDF statement "John has an ASU email ID" and a constraint "Only university students have an email account at university's server", the inference system should be able to infer "John is a student at ASU" and answer related queries accordingly.

5. RDF APPLICATIONS AND SEMANTIC WEB

The main purpose of the development of frameworks or models for metadata is to enable the development of a global Semantic Web, wherein different applications and Web sites can exchange information and hence exploit Web data to its full potential [53]. The current development of the Semantic Web is based on three complementary technologies:

1. XML allows users to define and use their own tags, thus various Web sites and applications can represent their metadata using such user defined schemas.
2. The semantics of the metadata is expressed using RDF. The RDF triples are written using XML tags; i.e, XML provides a syntactical framework.
3. Ontologies, or namespaces, are necessary to link the different collections and to compare or combine information across databases or applications. An ontology has a taxonomy and a set of inference rules. The taxonomy defines classes of objects and relations among them [29].

In this section, we will discuss current RDF applications and see how they use these three technologies to improve the semantic content on the Web.

Organizing the Web content and linking them based on content across different applications is the main idea behind Netscape's commercial product *Aurora* [8; 2]. Aurora is mainly an information management shell - a control place for managing information. On a traditional desktop, various resources are presented and accessed based on a protocol that does not consider the actual contents of the resources. Aurora tries to solve this shortcoming by providing a way of navigating various sources of data, such as FTP, local hard drive, and mailbox hierarchies, based on content. For example, a user can create a workspace called "Baseball" under which there can be some local files, baseball-related portions of sites like "Yahoo" and "CBSSports", some live search queries, and some emails, all related to *baseball*. Thus the data is stored on the basis of content and not on the basis of a content-neutral protocol. In other

words, Aurora provides the user with a way to personalize data organization.

Smart Browser [9; 14] takes this idea a step forward. Unlike a traditional browser, Smart Browser not only displays the requested information, but also provides additional related information, so that the user can access related data without much searching and browsing. The Browser creates its recommendations based on users' preferences and activities on the Web. For example if a user is looking at some tennis player's Website, the Browser may provide the user with details related to all Grand Slam tennis tournaments. The fundamental difference to the user is that access to data is no longer a bilateral relationship between the user and the content provider. It is a multilateral relation between the user, the content provider, and one or more third parties who are participating in the data access actively.

An RDF schema for providing a non-visual description of photos is defined in [33]. The proposed schema is expressed in three different parts:

- Dublin Core schema is used for identifying the photograph and describing properties like creator, editor, title etc.
- Technical schema is used for capturing technical data about the photo and the camera such as the type of camera, type of film, scanner and software used for digitization.
- Content schema is used for categorizing the subject of the photo by means of a controlled vocabulary. This schema allows photos to be retrieved based on such characteristics as portrait, group portrait, landscape, architecture, sport etc.

In addition to describing the content of Web pages in a structured way, RDF can also be used for representing the linkages between the Web pages, or for describing metadata about the current state and changes in documents on an entire Web site.

RDF has been used to create an extensible framework specifying user preferences and device capabilities [51]. Servers and content providers can use this information to describe user's preferences to customize the service or the content provided. For example, using computer hardware parameters, such as CPU and modem speed, the proper version of a Web page can be selected and presented to the user. Therefore, a general framework for content negotiation requires means for describing metadata, such as attributes and preferences of the user, attributes of the content, and rules for adapting content to the capabilities of the system and preferences of the user. Composite Capability/Preference Profile (CC/PP) [30], for example, is constructed as a two-level hierarchy of components (hardware and software platforms, and applications) and their attributes. Each component is described as a subtree whose branches are the capabilities or preferences associated with that component. A capability is described using CC/PP attributes, each having simple, atomic values.

The anticipation of a global Semantic Web also brings forth automated Web agents, which when assigned a particular task on the Web, will browse and search the complete Web and get the relevant information without any human interruption. Web agents will search for the various services available on the Web and using these services, will perform the tasks assigned by the user [6]. There are already a number of automated services available on the Web [6; 26], but there is no common language to describe a service in a way that lets agents understand the available function and take advantage of it. RDF can make this kind of service discovery process possible.

One of the ongoing projects to develop a Semantic Web aims at building an information management system on the present RDF mode [7]. In this project, the Web information is managed in two parts. Organizational information management, which includes the W3C Web information management framework [53], handles resource management and Web site access control. Personal information management, on the other hand, deals with individual documents and online information, and provides content- and context- based retrieval.

Another proposal toward developing a Semantic Web is *Metalog* [37], which provides a logical view of metadata present on the Web. Metalog allows users to write metadata, inference rules, and queries in simple English like language. The system further represents the reasoning rules both as RDF descriptions and as logic formulas.

6. FUTURE WORK AND CONCLUDING REMARKS

The success of a global semantic framework depends on our success in solving many practical challenges. Although these challenges are, from a technical point, mostly orthogonal to metadata description, storage, and retrieval problems, they will play an important role in the commercial success of the framework. Hence, they have to be considered with equal importance. Some of the challenges are listed below.

Security, for instance, presents itself as an important challenge when we consider about agents moving around the Web. As described earlier Web agents could browse the Web and get the required information and present it to the users. So a major issue is how trustworthy these Web agents are and what security constraints they have to follow. In some RDF applications, such as Smart Browsers, the user is concerned about how secure a third party is. The data obtained by the third party can be mishandled causing security and privacy problems. This becomes an especially serious concern when RDF used in electronic commerce or other business applications [21].

One type of security can be achieved by using *digital signatures and certificates*. Digital signatures provide authentication of the agent or the Web site providing the information. Like a written signature, the purpose of a digital signature is to guarantee that the individual (a user, an agent or a Web site) providing information really is who the individual claims to be. An RDF schema can be developed to hold security data. Whenever any access is made to the RDF model of an application, appropriate conditions could be validated with this security schema, before the access is granted.

Another practical challenge is to make RDF schemas easily available to describe various Web resources [12]. Currently, a number of resources are not well represented in RDF. One of the examples is email, which currently is one of the most important means of data exchange. Hence storing the metadata for emails can be useful for sorting out emails based on different criteria. Some of the property names that can be used in an RDF schema for email are *sender, date, subject, receiver, attachments*, and *mail format*. Once an RDF schema is available to describe a large number of property names in (at least) most common resources, RDF will achieve more widespread use. For example, RDF schemas should be implemented for representing non-textual files like image, audio, and video files. At present, little work has been done to handle the RDF representation of audio properties of a file. Similar to the photo management project [33] described earlier, technical schemas can be developed for audio and video files. This will help in representing audio files in a structured and uniform framework and will further help in querying and retrieving information from these files.

The basic purpose of developing a metadata description framework, like RDF, is to develop a global Semantic Web. At present, RDF is gaining its momentum and its applications are being developed and commercially validated. It opens up a new structured outlook toward the Web data, the organization of the Web data, and importance of metadata. If the Semantic Web is realized, it will have a major impact on how knowledge and information managed. But, both RDF and Semantic Web have long ways to go. At this stage, as this survey shows, most of the work are at their initial phases. Solutions to practical issues, such as security, ease of use, compatibility, will be crucial in the success of RDF. The conclusion drawn from the above is that the future of RDF is bright, its research and development opportunities are abundant, with RDF and XML, the next generation of the Web will be more organized, informative, searchable, accessible, and useful. We anticipate that the work and research of RDF and Semantic Web can significantly help in resource categorization, search and retrieval, data selection and reduction [36], therefore will inevitably impact on data mining and web mining, as well as on e-commerce and e-business.

7. REFERENCES

[1] Algae Howto. http://www.w3.org/1999/02/26-modules/User/Algae-HOWTO.html.

[2] Netscape, the Next Generation 'Aurora' Overview. http://home.netscape.com/browsers/future/aurora.html, 2000.

[3] R. Baeza-Yates and B. Ribeiro-Neto. *Morden Information Retrieval*. Addison Wesley and ACM Press, 1999.

[4] D. Beckett. The design and implementation of the redland RDF application framework. In *Proceedings of WWW10 conference*, 2001.

[5] T. Berners-Lee. The semantics toolbox: Building semantics on top of XML, 1998. http://www.w3.org/DesignIssues/Toolbox.html.

[6] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, May, 2001.

[7] T. Berners-Lee, D. Kerger, L. Andrea-Stein, R. Swick, and D. Weitzner. Proposal: Semantic Web Development, 2000. http://www.w3.org/2000/01/sw/developmentProposal.html.

[8] D. Brickley. Netscape, RDF: Aurora, 1999. http://www.mozilla.org/rdf/doc/aurora.html.

[9] D. Brickley. Rdf: Related links and other such fun stuff, 1999. http://www.mozilla.org/rdf/doc/SmartBrowsing.html.

[10] D. Brickley and R. Guha. Resource Description Framework (RDF) schema specification, 2000. http://www.w3.org/TR/RDF-schema.

[11] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. In *Proceedings of WWW7 Conference/Computer Networks and ISDN Systems*, pages 107–117, 1998.

[12] J. Broekstra, M. Klein, S. Decker, D. Fensen, F. Hamelen, and I. Horrocks. Enabling knowledge representation on the Web by extending rdf schema. In *Proceedings of WWW10 conference*, 2001.

[13] S. Chakrabarti. Data mining for hypertex: A tutorial survey. *SIGKDD explorations, ACM Newsletter of SIGKDD*, 1(2):1 – 11, 2000.

[14] M. Curtin and D. Ellison, G. ad Monroe. What's related? Everything but your privacy, 1999. http://www.interhack.net/pubs/whatsrelated.

[15] R. Daniel. A proposal for an RDF API, 1999. http://www.mailbase.ac.uk/lists/rdf-dev/1999-06/0002.html.

[16] Dublin Core Metadata Editor (DCDOT), 2000. http://www.ukoln.ac.uk/metadata/dcdot.

[17] S. Decker, D. Brickley, J. Saarela, and J. Angele. Query and inference service for RDF, 1998. http://purl.org/net/rdf/papers/QL98-queryservice-19981118.

[18] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and H. R. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391 – 407, 1990.

[19] Dublin Core Initiative And Metadata Element Set. http://dublincore.org/.

[20] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview. In U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 495–515. AAAI Press / The MIT Press, 1996.

[21] K. Forsberg and L. Dannstedt. Extensible use of RDF in business context. In *Proceedings of WWW9 conference*, 2000.

[22] R. Guha. rdfDB: An RDF database. http://web1.guha.com/rdfdb, 2000.

[23] R. Guha and T. Bray. Meta content framework using xml, 1997. http://www.w3.org/TR/NOTE-MCF-XML-970624.

[24] E. Harold and W. Means. *XML in A Nutshell*. O'Reilly, 2001.

[25] R. Heery. What is RDF. *Ariadne Magazine*, March, 1998.

[26] J. Hendler. Agents and semantic Web. *IEEE Computer*, April, 2001.

[27] R. Hoque. *XML for Real Programmers*. Moorgan Kaufmann, 2000.

[28] Jena - A Java API for RDF. http://www-uk.hpl.hp.com/people/bwm/rdf/jena/index.htm.

[29] C. Jenkins, M. Jackson, J. Burden, and J. Wallis. Automatic RDF Metadata Generation for Resource Discovery. In *Proceedings of WWW8 Conference*, pages 227 – 242, 1999.

[30] G. Klyne, F. Reynolds, C. Woodrow, and H. Ohto. Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies. W3C Working Draft, 2001. http://www.w3.org/TR/WD-CCPP-struct-vocab-20010315.html.

[31] S. Kokkelink. CARA Perl RDF-API-free software developed within the CARMEN project, 2000. http://cara.sourceforge.net/.

[32] R. Kosala and H. Blockeel. Web mining research: A survey. *SIGKDD explorations, ACM Newsletter of SIGKDD*, 2(1):1 – 15, 2000.

[33] Y. Lafon and B. Bos. Describing and Retrieving Photos using RDF, 2000. http://www.w3.org/TR/photo-rdf/.

[34] O. Lassila. Introduction to RDF metadata. W3C NOTE 1997-11-13, 1997. http://www.w3.org/TR/NOTE-rdf-simple-intro.

[35] O. Lassila and R. Swick. Resource Description Framework (RDF) model and syntax specification, 1999. http://www.w3.org/TR/REC-rdf-syntax.

[36] H. Liu and H. Motoda. Data reduction via instance selection. In H. Liu and H. Motoda, editors, *Instance Selection and Construction for Data Mining*, pages 3 – 20. Boston: Kluwer Academic Publishers, 2001.

[37] M. Marchiori and J. Saarela. Towards the Semantic Web, 1999. http://www.w3.org/RDF/Metalog/CIKM-050299.html.

[38] P. Margolis. *Computer & Internet Dictionary*. Random House Webster's, 1999.

[39] S. Melnik. Generic Interoperability Framework (GINF) project. http://www-diglib.stanford.edu/diglib/ginf/.

[40] S. Melnik. RDF API Draft. http://www-db.stanford.edu/ melnik/rdf/api.html.

[41] S. Melnik. RDF API 1.0 draft, 2000. http://www-db.stanford.edu/~ melnik/rdf/api.html.

[42] S. Melnik. Storing RDF in relational database, 2000. http://www-db.stanford.edu/~ melnik/rdf/db.html.

[43] E. Miller. An Introduction to the Resource Description Framework. *D-Lib Magazine*, May, 1998.

[44] L. Miller. Inkling architectural overview, 2001. http://ilrt.org/discovery/2001/07/inkling/.

[45] L. Miller. Inkling: RDF query using SquishQL, 2001. http://swordfish.rdfweb.org/rdfquery/.

[46] L. Page, S. Brin, R. Motwani, and W. T. The PageRank citation ranking: Bringing order to the Web. *Stanford University*, Draft, 1998.

[47] RDF.NET. http://injektilo.org/rdf/rdf.net.html.

[48] RDF technical overview. http://www.mozilla.org/rdf/doc/api.html.

[49] UKOLN, Web-focus RDF tools. http://www.ukoln.ac.uk/web-focus/events/seminars/what-is-rdf-may1998, May 1998.

[50] Resource discovery unit of DSTC, Reggie the metadata editor. http://metadata.net/dstc/SchemaFiles.html.

[51] F. Reynolds, J. Hjelm, S. Dawkins, and S. Singhal. Composite Capability/Preference Profiles (CC/PP): A User Side Framework for Content Negotiation, 1999. http://www.w3.org/TR/NOTE-CCPP.

[52] L. Seligman and A. Rosenthal. XML's impact on databases and data sharing. *Computer*, June:59 – 67, 2001.

[53] Web design issues: What a semantic Web can represent. http://www.w3.org/DesignIssues/RDFnot.html, May 1998.

[54] J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan. Web usage mining: Discovery and applications of usage pattern from web data. *SIGKDD explorations, ACM Newsletter of SIGKDD*, 1(2):12 – 23, 2000.

[55] XML Schemas, 2001. http://www.w3.org/XML/Schema.