Resource Description Framework (RDF) Schema Specification 1.0

W3C Candidate Recommendation 27 March 2000

This Version:
http://www.w3.org/TR/2000/CR-rdf-schema-20000327

Latest Version:
http://www.w3.org/TR/rdf-schema

Previous Version:
http://www.w3.org/TR/1999/PR-rdf-schema-19990303

Editors:
Dan Brickley, University of Bristol
R.V. Guha, Epinions

Acknowledgments

Copyright ©1998,1999,2000 W3C® (MIT, INRIA, Keio), All Rights Reserved. W3C liability, trademark, document use and software licensing rules apply.

Abstract

This specification describes how to use RDF to describe RDF vocabularies. The specification also defines a basic vocabulary for this purpose, as well as an extensibility mechanism to anticipate future additions to RDF.

Status of this document

This document is a Candidate Recommendation of the World Wide Web Consortium. Review comments on this specification should be sent by June 15, 2000 to <www-rdf-comments@w3.org>. The archive of public comments is available at http://lists.w3.org/Archives/Public/www-rdf-comments. Private comments that you wish to be visible only to the editors, working group chair, and W3C staff may be sent to <w3c-rdf-review@w3.org>.

This specification is a revision of the Proposed Recommendation of March 03 1999, incorporating editorial suggestions received in review comments. A separate document provides an overview of the main changes since the last publication of this work. With the publication of this document, the RDF Schema specification enters W3C Candidate Recommendation phase. W3C encourages active implementation to test this specification during the Candidate Recommendation review period; reports of implementation experience sent to the review address are especially desired.

The Resource Description Framework is part of the W3C Metadata Activity. The goal of this activity, and of RDF specifically, is to produce a language for the exchange of machine-understandable descriptions of resources on the Web. A separate specification describes the data model and syntax for the interchange of metadata using RDF.

This section describes the status of this document at the time of its publication. Other documents may supersede this document. Identified Errata in this document and the latest status of this document series is maintained at the W3C. Refer to Appendix B, About W3C Documents, for a description of the W3C Technical Report publishing policy.

It is inappropriate to use W3C Candidate Recommendations as reference material or to cite them as other than "work in
progress". This is work in progress and does not imply endorsement by, or the consensus of W3C. Advancement of a document to Candidate Recommendation is an explicit call for implementation and technical feedback; it is appropriate to build implementations based on this specification for the purposes of testing the specification and becoming familiar with it. A list of current W3C Recommendations and other technical documents can be found at http://www.w3.org/TR.

Table of Contents

● 1. Introduction
  ○ 1.1. Scope
    ■ 1.1.1. Platform for Internet Content Selection (PICS)
    ■ 1.1.2. Simple Web Metadata
    ■ 1.1.3. Sitemaps and Concept Navigation

● 2. Classes and Properties
  ○ 2.1. The Type System
    ■ 2.1.1. Figure 1: Classes and Resources as Sets and Elements
    ■ 2.1.2. Figure 2: Class hierarchy for the RDF Schema
  ○ 2.2. Core Classes
    ■ 2.2.1. rdfs:Resource
    ■ 2.2.2. rdf:Property
    ■ 2.2.3. rdfs:Class
  ○ 2.3. Core Properties
    ■ 2.3.1. rdf:type
    ■ 2.3.2. rdfs:subClassOf
    ■ 2.3.3. rdfs:subPropertyOf
    ■ 2.3.4. rdfs:seeAlso
    ■ 2.3.5. rdfs:isDefinedBy

● 3. Constraints
  ○ 3.1. Core Constraints
    ■ 3.1.1. rdfs:ConstraintResource
    ■ 3.1.2. rdfs:ConstraintProperty
    ■ 3.1.3. rdfs:range
    ■ 3.1.4. rdfs:domain
  ○ 3.2 Example

● 4. Extensibility Mechanisms
  ○ 4.1 Evolvability of RDF vocabularies
    ■ 4.1.1. Terminology
    ■ 4.1.2. Versioning and URI references
    ■ 4.1.3. Inter-Vocabulary Relationships
  ○ 4.2. Evolvability of the RDF Schema Constraint Mechanism

● 5. Documentation
  ○ 5.1. rdfs:comment
  ○ 5.2. rdfs:label

● 6. Model and Syntax concepts
  ○ 6.1. rdfs:Literal
RDF Schemas

1. Introduction

The Resource Description Framework (RDF) is a foundation for processing metadata; it provides interoperability between applications that exchange machine-understandable information on the Web. RDF uses XML to exchange descriptions of Web resources but the resources being described can be of any type, including XML and non-XML resources. RDF emphasizes facilities to enable automated processing of Web resources. RDF can be used in a variety of application areas, for example: in resource discovery to provide better search engine capabilities, in cataloging for describing the content and content relationships available at a particular Web site, page, or digital library, by intelligent software agents to facilitate knowledge sharing and exchange, in content rating, in describing collections of pages that represent a single logical "document", for describing intellectual property rights of Web pages, and for expressing the privacy preferences of a user as well as the privacy policies of a Web site. RDF with digital signatures will be key to building the "Web of Trust" for electronic commerce, collaboration, and other applications.

Descriptions used by these applications can be modeled as relationships among Web resources. The RDF data model, as specified in [RDFMS], defines a simple model for describing interrelationships among resources in terms of named properties and values. RDF properties may be thought of as attributes of resources and in this sense correspond to traditional attribute-value pairs. RDF properties also represent relationships between resources. As such, the RDF data model can therefore resemble an entity-relationship diagram. The RDF data model, however, provides no mechanisms for declaring these properties, nor does it provide any mechanisms for defining the relationships between these properties and other resources. That is the role of RDF Schema.

Resource description communities require the ability to say certain things about certain kinds of resources. For describing bibliographic resources, for example, descriptive attributes including "author", "title", and "subject" are common. For digital certification, attributes such as "checksum" and "authorization" are often required. The declaration of these properties (attributes) and their corresponding semantics are defined in the context of RDF as an RDF schema. A schema defines not only the properties of the resource (e.g., title, author, subject, size, color, etc.) but may also define the kinds of resources being described (books, Web pages, people, companies, etc.).

This document does not specify a vocabulary of descriptive elements such as "author". Instead, it specifies the mechanisms needed to define such elements, to define the classes of resources they may be used with, to restrict possible combinations of
classes and relationships, and to detect violations of those restrictions. Thus, this document defines a schema specification language. More succinctly, the RDF Schema mechanism provides a basic type system for use in RDF models. It defines resources and properties such as rdfs:Class and rdfs:subClassOf that are used in specifying application-specific schemas.

The typing system is specified in terms of the basic RDF data model - as resources and properties. Thus, the resources constituting this typing system become part of the RDF model of any description that uses them. The schema specification language is a declarative representation language influenced by ideas from knowledge representation (e.g., semantic nets, frames, predicate logic) as well as database schema specification languages (e.g. NIAM) and graph data models. The RDF schema specification language is less expressive, but much simpler to implement, than full predicate calculus languages such as CycL and KIF.

RDF and the RDF Schema language were also based on metadata research in the Digital Library community. In particular, RDF adopts a modular approach to metadata that can be considered an implementation of the Warwick Framework [WF]. RDF represents an evolution of the Warwick Framework model so that the Warwick Framework allowed each metadata vocabulary to be represented in a different syntax. In RDF, all vocabularies are expressed within a single well defined model. This allows for a finer grained mixing of machine-processable vocabularies, and addresses the need to create metadata in which statements can draw upon multiple vocabularies that are managed in a decentralized fashion by independent communities of expertise.

RDF Schemas might be contrasted with XML Document Type Definitions (DTDs) and XML Schemas. Unlike an XML DTD or Schema, which gives specific constraints on the structure of an XML document, an RDF Schema provides information about the interpretation of the statements given in an RDF data model. While an XML Schema can be used to validate the syntax of an RDF/XML expression, a syntactic schema alone is not sufficient for RDF purposes. RDF Schemas may also specify constraints that should be followed by these data models. Future work on RDF Schema and XML Schema might enable the simple combination of syntactic and semantic rules from both [SCHEMA-ARCH].

This RDF Schema specification has intentionally left unspecified a set of primitive datatypes. As RDF uses XML for its interchange encoding, the work on data typing in XML itself should be the foundation for such a capability.

1.1. Scope

The RDF Schema specification is not aimed at theoretical issues, but at solving a small number of immediate problems. Its creators expect that other problems (some of which are illustrated in the examples below) will share similar characteristics and that they also may be able to use the basic classes described in this specification.

The RDF Schema specification was directly influenced by consideration of the following problems:

1.1.1. Platform for Internet Content Selection (PICS)

The RDF Model and Syntax is adequate to represent PICS labels, however it does not provide a general-purpose mapping from PICS rating systems into an RDF representation. One such mapping is described in a separate document.

1.1.2. Simple Web Metadata

One obvious application for RDF is in the description of Web pages. This is one of the basic goals of the Dublin Core Metadata Initiative. The Dublin Core Element Set is a set of 15 elements believed to be broadly applicable to describing Web resources to enable their discovery. The Dublin Core has been a major influence on the development of RDF. An important consideration in the development of the Dublin Core was to not only allow simple descriptions, but also to provide the ability to qualify descriptions in order to provide both domain specific elaboration and descriptive precision.

The RDF Schema Specification provides a machine-understandable system for defining schemas for descriptive vocabularies like the Dublin Core. It allows designers to specify classes of resource types and properties to convey descriptions of those classes, relationships between those properties and classes, and constraints on the allowed combinations of classes, properties, and values.

1.1.3. Sitemaps and Concept Navigation

A sitemap is a hierarchical description of a Web site. A subject taxonomy is a classification system that might be used by content creators or trusted third parties to organize or classify Web resources. The RDF Schema specification provides a
mechanism for defining the vocabularies needed for such applications.

Thesauri and library classification schemes are well known examples of hierarchical systems for representing subject taxonomies in terms of the relationships between named concepts. The RDF Schema specification provides sufficient resources for creating RDF models that represent the logical structure of thesauri (and other library classification systems).

1.1.4. P3P

The W3C Platform for Privacy Preferences Project (P3P) has specified a grammar for constructing statements about a site’s data collection practices and personal preferences as exercised over those practices, as well as a syntax for exchanging structured data.

Although personal data collection practices have been described in P3P using an application-specific XML tagset, there are benefits to using a general metadata model for this data. The structure of P3P policies can be interpreted as an RDF model. Using a metadata schema to describe the semantics of privacy practice descriptions will permit privacy practice data to be used along with other metadata in a query during resource discovery, and will permit a generic software agent to act on privacy metadata using the same techniques as used for other descriptive metadata. Extensions to P3P that describe the specific data elements collected by a site could use RDF Schema to further specify how those data elements are used.

2. Classes and Properties

An RDF Schema is expressed by the data model described in the RDF Model and Syntax [RDFMS] specification. The schema description language is simply a set of resources and properties defined by the RDF Schema Specification and implicitly part of every RDF model using the RDF schema machinery.

This document specifies the RDF Schema mechanism as a set of RDF resources (including classes and properties), and constraints on their relationships. The abstract RDF Schema core vocabulary can be used to make RDF statements defining and describing application-specific vocabularies such as the Dublin Core Element Set.

2.1. The Type System

The RDF Schema defined in this specification is a collection of RDF resources that can be used to describe properties of other RDF resources (including properties) which define application-specific RDF vocabularies. The core schema vocabulary is defined in a namespace informally called 'rdfs' here, and identified by the URI reference http://www.w3.org/2000/01/rdf-schema#. This specification also uses the prefix 'rdf' to refer to the core RDF namespace http://www.w3.org/1999/02/22-rdf-syntax-ns#.

As described in the RDF Model and Syntax specification [RDFMS], resources may be instances of one or more classes; this is indicated with the rdf:type property. Classes themselves are often organized in a hierarchical fashion, for example a class Dog might be considered a subclass of Mammal which is a subclass of Animal, meaning that any resource which is of rdf:type Dog is also considered to be of rdf:type Animal. This specification describes a property, rdfs:subClassOf, to denote such relationships between classes.

The RDF Schema type system is similar to the type systems of object-oriented programming languages such as Java. However, RDF differs from many such systems in that instead of defining a class in terms of the properties its instances may have, an RDF schema will define properties in terms of the classes of resource to which they apply. This is the role of the rdfs:domain and rdfs:range constraints described in Section 3. For example, we could define the author property to have a domain of Book and a range of Literal, whereas a classical OO system might typically define a class Book with an attribute called author of type Literal. One benefit of the RDF property-centric approach is that it is very easy for anyone to say anything they want about existing resources, which is one of the architectural principles of the Web [BERNERS-LEE98].

This specification anticipates the development of a set of classes corresponding to a set of datatypes. This specification does not define any specific datatypes, but does note that datatypes may be used as the value of the rdfs:range property.

2.1.1. Figure 1: Classes and Resources as Sets and Elements

Figure 1 illustrates the concepts of class, subclass, and resource. A class is depicted by a rounded rectangle; a resource is depicted by a large dot. In the figure below, arrows are drawn from a resource to the class it defines. A subclass is shown by having a rounded rectangle (the subclass) completely enclosed by another (the superclass). If a resource is inside a class, then there exists either an explicit or implicit rdf:type property of that resource whose value is the resource defining the
containing class. (These properties are shown as arcs in the directed labelled graph representation in figure 2). The RDF resources depicted in figure 1 are described either in the remainder of this specification, or in the RDF Model and Syntax specification.

Figure 1: Classes and Resources as Sets and Elements

2.1.2. Figure 2: Class hierarchy for the RDF Schema

Figure 2 shows the same information about the class hierarchy as in figure 1, but does so using a "nodes and arcs" graph representation of the RDF data model. If one class is a subset of another, then there is an rdfs:subClassOf arc from the node representing the first class to the node representing the second. Similarly, if a resource is an instance of a class, then there is an rdf:type arc from the resource to the node representing the class. Not all such arcs are shown. We only show the arc to the most tightly encompassing class, and rely on the transitivity of the rdfs:subClassOf relation to provide the rest.

Figure 2: Class Hierarchy for the RDF Schema
2.2. Core Classes

The following resources are the core classes that are defined as part of the RDF Schema vocabulary. Every RDF model that draws upon the RDF Schema namespace (implicitly) includes these.

2.2.1. rdfs:Resource

All things being described by RDF expressions are called resources, and are considered to be instances of the class rdfs:Resource. The RDF class rdfs:Resource represents the set called 'Resources' in the formal model for RDF presented in section 5 of the Model and Syntax specification [RDFMS].

2.2.2. rdf:Property

rdf:Property represents the subset of RDF resources that are properties, i.e., all the elements of the set introduced as 'Properties' in section 5 of the Model and Syntax specification [RDFMS].

2.2.3. rdfs:Class

This corresponds to the generic concept of a Type or Category, similar to the notion of a Class in object-oriented programming languages such as Java. When a schema defines a new class, the resource representing that class must have an rdf:type property whose value is the resource rdfs:Class. RDF classes can be defined to represent almost anything, such as Web pages, people, document types, databases or abstract concepts.

2.3. Core Properties

Every RDF model which uses the schema mechanism also (implicitly) includes the following core properties. These are instances of the rdf:Property class and provide a mechanism for expressing relationships between classes and their instances or superclasses.

2.3.1. rdf:type

This indicates that a resource is a member of a class, and thus has all the characteristics that are to be expected of a member of that class. When a resource has an rdf:type property whose value is some specific class, we say that the resource is an instance of the specified class. The value of an rdf:type property for some resource is another resource which must be an instance of rdfs:Class. The resource known as rdfs:Class is itself a resource of rdf:type rdfs:Class. Individual classes (for example, 'Dog') will always have an rdf:type property whose value is rdfs:Class (or some subclass of rdfs:Class, as described in section 2.3.2). A resource may be an instance of more than one class.

2.3.2. rdfs:subClassOf

This property specifies a subset/superset relation between classes. The rdfs:subClassOf property is transitive. If class A is a subclass of some broader class B, and B is a subclass of C, then A is also implicitly a subclass of C. Consequently, resources that are instances of class A will also be instances of C, since A is a sub-set of both B and C. Only instances of rdfs:Class can have the rdfs:subClassOf property and the property value is always of rdf:type rdfs:Class. A class may be a subclass of more than one class.

A class can never be declared to be a subclass of itself, nor of any of its own subclasses. Note that this constraint is not expressible using the RDF Schema constraint facilities provided below, and so does not appear in the RDF version of this specification given in Appendix A.

2.3.2.1. Example

This is a very simple example that expresses the following class hierarchy. We first define a class MotorVehicle. We then define three subclasses of MotorVehicle, namely PassengerVehicle, Truck and Van. We then define a class Minivan which is a subclass of both Van and PassengerVehicle.
The RDF/XML shown here uses the basic RDF syntax defined in section 2.2.1 of the Model and Syntax specification [RDFMS]. abbreviation mechanism provided by the RDF serialization syntax.

```
<rdf:RDF xml:lang="en"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <!-- Note: this RDF schema would typically be used in RDF instance data by referencing it with an XML namespace declaration, for example xmlns:xyz="http://www.w3.org/2000/03/example/vehicles#". This allows us to use abbreviations such as xyz:MotorVehicle to refer unambiguously to the RDF class 'MotorVehicle'. -->

  <rdf:Description ID="MotorVehicle">
    <rdf:type resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
  </rdf:Description>

  <rdf:Description ID="PassengerVehicle">
    <rdf:type resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
  </rdf:Description>

  <rdf:Description ID="Truck">
    <rdf:type resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
  </rdf:Description>

  <rdf:Description ID="Van">
    <rdf:type resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
  </rdf:Description>

  <rdf:Description ID="MiniVan">
    <rdf:type resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="#Van"/>
    <rdfs:subClassOf rdf:resource="#PassengerVehicle"/>
  </rdf:Description>
</rdf:RDF>
```
2.3.3. rdfs:subPropertyOf

The property rdfs:subPropertyOf is an instance of rdf:Property that is used to specify that one property is a specialization of another. A property may be a specialization of zero, one or more properties. If some property P2 is a subPropertyOf another more general property P1, and if a resource A has a P2 property with a value B, this implies that the resource A also has a P1 property with value B.

A property can never be declared to be a subproperty of itself, nor of any of its own subproperties. Note that this constraint is not expressible using the RDF Schema constraint facilities provided below, and so does not appear in the RDF version of this specification given in Appendix A.

2.3.3.1. Example

If the property biologicalFather is a subproperty of the broader property biologicalParent, and if Fred is the biologicalFather of John, then it is implied that Fred is also the biologicalParent of John.

```xml
<rdf:RDF xml:lang="en"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <rdf:Description ID="biologicalParent">
    <rdf:type resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  </rdf:Description>

  <rdf:Description ID="biologicalFather">
    <rdf:type resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:subPropertyOf rdf:resource="#biologicalParent"/>
  </rdf:Description>
</rdf:RDF>
```

2.3.4. rdfs:seeAlso

The property rdfs:seeAlso specifies a resource that might provide additional information about the subject resource. This property may be specialized using rdfs:subPropertyOf to more precisely indicate the nature of the information the object resource has about the subject resource. The object and the subject resources are constrained only to be instances of the class rdfs:Resource.

2.3.5. rdfs:isDefinedBy

The property rdfs:isDefinedBy is a subproperty of rdfs:seeAlso, and indicates the resource defining the subject resource. As with rdfs:seeAlso, this property can be applied to any instance of rdfs:Resource and may have as its value any rdfs:Resource.

The most common anticipated usage is to identify an RDF schema, given a name for one of the properties or classes defined by that schema. Although XML namespace declarations will typically provide the URI where RDF vocabulary resources are defined, there are cases where additional information is required.

For example, constructs such as

```xml
<rdfs:subPropertyOf rdf:resource="http://purl.org/dc/elements/1.0/Creeator"/>
```

do not indicate the URI of the schema that includes the vocabulary item Creator (i.e., http://purl.org/dc/elements/1.0/).

In such cases, the rdfs:isDefinedBy property can be used to explicitly represent that information. This approach will also work when the URIs of the namespace and its components have no obvious relationship, as would be the case if they were identified using schemes such as GUIDs or MD-5 hashes.
3. Constraints

This specification introduces an RDF vocabulary for making statements about constraints on the use of properties and classes in RDF data. For example, an RDF schema might describe limitations on the types of values that are valid for some property, or on the classes to which it makes sense to ascribe such properties.

RDF Schema provides a mechanism for describing such constraints, but does not say whether or how an application must process the constraint information. For example, while an RDF schema can assert that an author property is used to indicate resources that are members of the class Person, it does not say whether or how an application should act in processing that class information. We expect that different applications will use these constraints in different ways - e.g., a validator will look for errors, an interactive editor might suggest legal values, and a reasoning application might infer the class and then announce any inconsistencies.

RDF schemas can express constraints that relate vocabulary items from multiple independently developed schemas. Since URI references are used to identify classes and properties, it is possible to create new properties whose domain or range constraints reference classes defined in another namespace.

The following constraints are specified in RDF Schema 1.0: rdfs:domain and rdfs:range constraints on property usage, the rule that rdfs:subPropertyOf and rdfs:subClassOf properties should not form loops, plus any further constraints defined using the rdfs:ConstraintResource extensibility mechanism. Different applications may exhibit different behaviors when dealing with RDF constraints.

Some examples of constraints include:

- That the value of a property should be a resource of a designated class. This is known as a range constraint. For example, a range constraint applying to the author property might express that the value of an author property must be a resource of class Person.
- That a property may be used on resources of a certain class. This is known as a domain constraint. For example, that the author property could only originate from a resource that was an instance of class Book.

This specification does not attempt to enumerate every possible form of constraint applicable to RDF vocabulary description. Instead, some basic constraint mechanisms are defined here, accompanied by an extension facility to allow for the subsequent additions of new types of constraint.

Although the RDF data model does not allow for explicit properties (such as an rdf:type property) to be ascribed to Literals (atomic values), we nevertheless consider these entities to be members of classes (e.g., the string "John Smith" is considered to be a member of the class rdfs:Literal.)

Note: We expect future work in RDF and XML data-typing to provide clarifications in this area.

3.1. Core Constraints

3.1.1. rdfs:ConstraintResource

This resource defines a subclass of rdfs:Resource whose instances are RDF schema constructs involved in the expression of constraints. The purpose of this class is to provide a mechanism that allows RDF processors to assess their ability to use the constraint information associated with an RDF model. Since this specification does not provide a mechanism for the dynamic discovery of new forms of constraint, an RDF Schema 1.0 processor encountering previously unknown instances of rdfs:ConstraintResource can be sure that it is unqualified to determine the meaning of those constraints.

3.1.2. rdfs:ConstraintProperty

This resource defines a subclass of rdf:Property, all of whose instances are properties used to specify constraints. This class is a subclass of rdfs:ConstraintResource and corresponds to the subset of that class representing properties. Both rdfs:domain and rdfs:range are instances of rdfs:ConstraintProperty.

3.1.3. rdfs:range

An instance of ConstraintProperty that is used to indicate the class(es) that the values of a property must be members of. The value of a range property is always a Class. Range constraints are only applied to properties.

A property can have at most one range property. It is possible for it to have no range, in which case the class of the property
value is unconstrained.

**Constraints on rdfs:range**

The rdfs:domain of rdfs:range is the class rdf:Property. This indicates that the range property applies to resources that are themselves properties.

The rdfs:range of rdfs:range is the class rdfs:Class. This indicates that any resource that is the value of a range property will be a class.

**Note: specifying multiple range classes**

Although it is not permitted to express two or more range constraints on a property, a similar outcome can be achieved by defining a common superclass for any classes that represent appropriate values for some property. For example, to express the constraint that a property xyz:drivesMotorVehicle can have values which are Vans, Trucks or PassengerVehicles, we assert that xyz:drivesMotorVehicle has a rdfs:range of MotorVehicle. If Van, Truck and PassengerVehicle are known to be subclasses of MotorVehicle, then all these types of resource are acceptable values for xyz:drivesMotorVehicle. In cases where a common superclass does not exist, one can be defined in a schema in the normal manner.

**3.1.4. rdfs:domain**

An instance of ConstraintProperty that is used to indicate the class(es) on whose members a property can be used.

A property may have zero, one, or more than one class as its domain. If there is no domain property, it may be used with any resource. If there is exactly one domain property, it may only be used on instances of that class (which is the value of the domain property). If there is more than one domain property, the constrained property can be used with instances of any of the classes (that are values of those domain properties).

**Constraints on rdfs:domain**

The rdfs:domain of rdfs:domain is the class rdf:Property. This indicates that the domain property is used on resources that are properties.

The rdfs:range of rdfs:domain is the class rdfs:Class. This indicates that any resource that is the value of a domain property will be a class.

Note: This specification does not constraint the number of rdfs:domain properties that a property may have. If there is no domain property, we know nothing about the classes with which the property is used. If there is more than one rdfs:domain property, the constrained property can be used with resources that are members of any of the indicated classes. Note that unlike range this is a very weak constraint.

**Figure 4: Constraints in the RDF Schema**

The RDF Schema uses the constraint properties to constrain how its own properties can be used. These constraints are shown below in figure 4. Nodes with **bold** outlines are instances of rdfs:Class.
3.2. Example

Continuing with our earlier example of MotorVehicle, in this example, we define two properties: registeredTo and rearSeatLegRoom. The registeredTo property is applicable to any MotorVehicle and its value is a Person (defined in the examples below). For the sake of this example, rearSeatLegRoom only applies to Minivans and PassengerVehicles. The value is a Number (we anticipate that some concept like this will be provided by future work on data types), which is the number of centimeters of rear seat legroom.

```xml
<rdf:RDF xml:lang="en"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <rdf:Description ID="registeredTo">
        <rdf:type resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
        <rdfs:domain rdf:resource="#MotorVehicle"/>
        <rdfs:range rdf:resource="#Person"/>
    </rdf:Description>

    <rdf:Description ID="rearSeatLegRoom">
        <rdf:type resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
        <rdfs:domain rdf:resource="#PassengerVehicle"/>
        <rdfs:domain rdf:resource="#Minivan"/>
        <rdfs:range rdf:resource="http://www.w3.org/2000/03/example/classes#Number"/>
    </rdf:Description>
</rdf:RDF>
```

4. Extensibility Mechanisms

The RDF Schema specification builds upon the foundations provided by XML and by the RDF Model and Syntax. It provides some additional facilities to support the evolution both of individual RDF vocabularies, and of the core RDF Schema specification vocabulary introduced in this document.

4.1. Evolvability of RDF vocabularies

The Resource Description Framework is intended to be flexible and easily extensible; this suggests that a great variety of schemas will be created and that new and improved versions of these schemas will be a common occurrence on the Web.
4.1.1. Terminology

The phrase 'RDF vocabulary' is used here to refer to those resources which evolve over time; 'RDF schema' is used to denote those resources which constitute the particular (unchanging) versions of an RDF vocabulary at any point in time. Thus we might talk about the evolution of the Dublin Core vocabulary. Each version of the Dublin Core vocabulary would be a different RDF schema, and would have a corresponding RDF model and concrete syntactic representation.

4.1.2. Versioning and URI references

RDF uses the XML Namespace facility [XMLNS] to identify the schema in which the properties and classes are defined. Since changing the logical structure of a schema risks breaking other RDF models which depend on that schema, this specification recommends that a new namespace URI should be declared whenever an RDF schema is changed.

In effect, changing the RDF statements which constitute a schema creates a new one; new schema namespaces should have their own URI to avoid ambiguity. Since an RDF Schema URI unambiguously identifies a single version of a schema, software that uses or manages RDF (eg., caches) should be able to safely store copies of RDF schema models for an indefinite period. The problems of RDF schema evolution share many characteristics with XML DTD version management and the general problem of Web resource versioning. A general approach to these issues is beyond the scope of this specification.

Since each RDF schema has its own unchanging URI, these can be used to construct unique URI references for the resources defined in a schema. This is achieved by combining the local identifier for a resource with the URI associated with that schema namespace. The XML representation of RDF uses the XML namespace mechanism for associating elements and attributes with URI references for each vocabulary item used.

4.1.3. Inter-Vocabulary Relationships

The resources defined in RDF schemas are themselves Web resources, and can be described in other RDF schemas. This principle provides the basic mechanism for RDF vocabulary evolution. This specification does not attempt to provide a full framework for expressing mappings between schemas; it does however provide the rdfs:subClassOf and rdfs:subPropertyOf properties. The ability to express specialization relationships between classes (subClassOf) and between properties (subPropertyOf) provides a simple mechanism for making statements about how such resources map to their predecessors.

There are many scenarios for which these simple mechanisms are not adequate; a more comprehensive schema mapping mechanism for RDF may be developed in future W3C Activity.

4.1.3.1. Examples

A schema representing version 1.0 of some vocabulary might define classes corresponding to a number of vehicle types. The schema for version 2.0 of this vocabulary constitutes a different Web resource. If the new schema defines for example a class 'Van' whose members are a subset of the members of the class 'Van' in version 1.0, the rdfs:subClassOf property can be used to state that all instances of V2:Van are also instances of V1:Van.

Where the vocabulary defines properties, the same approach can be taken, using rdfs:subPropertyOf to make statements about relationships between properties defined in successive versions of an RDF vocabulary.

4.2. Evolvability of the RDF Schema Constraint Mechanism

This specification defines a subclass of resources known as 'constraint resources' (section 3.1). This is provided to allow for the addition of new ways of expressing RDF constraints. Future extensions to the Resource Description Framework may introduce new resources that are instances of the rdfs:ConstraintResource class. It is necessary to anticipate RDF content which draws upon properties or classes defined using constraints other than those available in this version of RDF. As yet unknown constraints may contribute to a more expressive framework for specifying RDF constraints.

RDF agents unfamiliar with the semantics of unknown instances of rdfs:ConstraintResource may therefore lack the knowledge to evaluate constraint satisfaction when vocabulary items are defined using those unknown constraints. Since RDF itself may not represent declaratively the full meaning of these constraint resources, the acquisition of RDF statements about a new ConstraintResource may not provide enough information to enable its use. For example, when encountering a previously unknown constraint property type called RDF3:mysteryConstraint we may learn from a schema that it has a range of rdfs:Class and a domain of rdf:Property. The range and domain constraints if encountered alone would be enough to tell us how to legally use RDF3:mysteryConstraint, but they do not tell us anything about the nature of the constraint expressed when it is used in that fashion.
The `rdfs:ConstraintResource` construct is provided here as a simple future-proofing mechanism, and addresses some of the issues discussed at greater length in the Extensible Web Languages W3C NOTE [EXTWEB]. By flagging new forms of constraint as members of this class, we indicate that they are intended to express RDF Schema language constraints whose semantics must be understood for constraint checking to be possible.

Membership in the `rdfs:ConstraintResource` class suggests, but does not imply, that those semantics may be inexpressible in a declarative form. Since the expressive facilities available within RDF for doing so are also likely to evolve, this distinction itself presents a moving target. All RDF agents will have implicit knowledge of certain constraints (for example, this specification declares that `subClassOf` properties must not form a loop in an RDF graph) which may or may not be capable of representation within (some version of) RDF. It may be the case that some future RDF specification provides facilities which will allow RDF agents to comprehend declarative specifications for as-yet uninvented constraint properties. In such a case, these agents could safely comprehend (some) previously unencountered forms of constraint. By providing the basic `rdfs:ConstraintResource` class, we anticipate such developments. All RDF agents written solely to this specification will appreciate their ignorance of the meaning of unknown instances of that class, since this specification provides no mechanism for learning about such constraints through the interpretation of RDF statements. Future specifications, should they offer such facilities, could also define subclasses of `ConstraintProperty` to classify new constructs according to whether or not they had inexpressible semantics.

5. Documentation

The following properties are provided to support simple documentation and user-interface related annotations within RDF schemas. Multilingual documentation of schemas is supported at the syntactic level through use of the `xml:lang` language tagging facility. Since RDF schemas are expressed within the RDF data model, vocabularies defined in other namespaces may be used to provide richer documentation.

5.1. `rdfs:comment`

This is used to provide a human-readable description of a resource.

5.2. `rdfs:label`

This is used to provide a human-readable version of a resource name.

6. Model and Syntax concepts

The RDF Model and Syntax specification [RDFMS] introduces the base concepts of RDF. A number of these are defined formally in the RDF Schema whose namespace URI is `http://www.w3.org/1999/02/22-rdf-syntax-ns#`. In addition, some further concepts are introduced in the RDF Model and Syntax specification but do not appear in the RDF Model and Syntax schema. These formally belong in the Schema namespace (for example, `rdfs:Literal` and `rdfs:Resource`). In cases where an RDF resource belongs to the `http://www.w3.org/1999/02/22-rdf-syntax-ns#` namespace, this document can provide only a convenience copy of that resource's definition.

Appendix A provides an RDF/XML schema for the RDF resources defined in this document, including RDF Model concepts such as `Literal` and `Resource`. The RDF/XML Schema in Appendix A also makes RDF statements about resources defined in the RDF Model and Syntax namespace. These have the status of annotations rather than definitions.

6.1. `rdfs:Literal`

This corresponds to the set called the 'Literals' in the formal model for RDF presented in section 5 of the Model and Syntax specification [RDFMS]. Atomic values such as textual strings are examples of RDF literals.

6.2. `rdf:Statement`

This corresponds to the set called the 'Statement' in the formal model for RDF presented in section 5 of the Model and Syntax specification [RDFMS].

6.3. `rdf:subject`
This corresponds to the property called the 'subject' in the formal model for RDF presented in section 5 of the Model and Syntax specification [RDFMS]. Its rdfs:domain is rdf:Statement and rdfs:range is rdfs:Resource. This is used to specify the resource described by a reified statement.

6.4. rdf:predicate

This corresponds to the property called the 'predicate' in the formal model for RDF presented in section 5 of the Model and Syntax specification [RDFMS]. Its rdfs:domain is rdf:Statement and rdfs:range is rdf:Property. This is used to identify the property used in the modeled statement.

6.5. rdf:object

This corresponds to the property called the 'object' in the formal model for RDF presented in section 5 of the Model and Syntax specification [RDFMS]. Its rdfs:domain is rdf:Statement. This is used to identify the property value in the modeled statement.

6.6. rdfs:Container

This class is used to represent the Container classes described in section 3 of the Model and Syntax specification [RDFMS]. It is an instance of rdfs:Class and rdfs:subClassOf of rdfs:Resource.

6.7. rdf:Bag

This corresponds to the class called 'Bag' in the formal model for RDF presented in section 5 of the Model and Syntax specification [RDFMS]. It is an instance of rdfs:Class and rdfs:subClassOf rdfs:Container.

6.8. rdf:Seq

This corresponds to the class called 'Sequence' in the formal model for RDF presented in section 5 of the Model and Syntax specification [RDFMS]. It is an instance of rdfs:Class and rdfs:subClassOf rdfs:Container.

6.9. rdf:Alt

This corresponds to the class called 'Alternative' in the formal model for RDF presented in section 5 of the Model and Syntax specification [RDFMS]. It is an instance of rdfs:Class and rdfs:subClassOf rdfs:Container.

6.10. rdfs:ContainerMembershipProperty

This class has as members the properties _1, _2, _3 ... used to indicate container membership, as described in section 3 of the Model and Syntax specification [RDFMS]. This is a rdfs:subClassOf rdf:Property.

6.11. rdf:value

This corresponds to the 'value' property described in section 2.3 of the Model and Syntax specification [RDFMS].

7. Examples

This section gives some brief examples of using the RDF Schema machinery to define classes and properties for some possible applications. Note that some of these examples use the abbreviated RDF syntax (mentioned in 2.3.2.1 above) to express class membership.

7.1. Example 1

In this example, Person is a class with a corresponding human-readable description of "The class of people". Animal is a class presumed to be defined in another schema. All persons are animals, so we declare that Person is a subclass of Animal. A Person may have an age property. The value of age is an integer. A Person may also have an ssn ("Social Security Number") property. The value of ssn is an integer. A Person's marital status is one of {Single, Married, Divorced, Widowed}. This is achieved through use of the rdfs:range constraint: we define both a maritalStatus property and a class MaritalStatus (adopting the convention of using lower case letters to begin the names of properties, and capitals for
classes). We then use rdfs:range to state that a maritalStatus property only 'makes sense' when it has a value which is an instance of the class MaritalStatus. The schema then defines a number of instances of this class. Whether resources declared to be of type MaritalStatus in another graph are trusted is an application level decision.

```
<rdf:RDF xml:lang="en"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <rdfs:Class rdf:ID="Person">
    <rdfs:comment>The class of people.</rdfs:comment>
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/03/example/classes#Animal"/>
  </rdfs:Class>

  <rdf:Property ID="maritalStatus">
    <rdfs:range rdf:resource="#MaritalStatus"/>
    <rdfs:domain rdf:resource="#Person"/>
  </rdf:Property>

  <rdf:Property ID="ssn">
    <rdfs:comment>Social Security Number</rdfs:comment>
    <rdfs:range rdf:resource="http://www.w3.org/2000/03/example/classes#Integer"/>
    <rdfs:domain rdf:resource="#Person"/>
  </rdf:Property>

  <rdf:Property ID="age">
    <rdfs:range rdf:resource="http://www.w3.org/2000/03/example/classes#Integer"/>
    <rdfs:domain rdf:resource="#Person"/>
  </rdf:Property>

  <rdfs:Class rdf:ID="MaritalStatus"/>
  <MaritalStatus rdf:ID="Married"/>
  <MaritalStatus rdf:ID="Divorced"/>
  <MaritalStatus rdf:ID="Single"/>
  <MaritalStatus rdf:ID="Widowed"/>
</rdf:RDF>
```

7.2. Example 2

In this example we sketch an outline of an RDF vocabulary for use with searchable Internet services. SearchQuery is declared to be a class. Every SearchQuery can have both a queryString whose value is an rdfs:Literal and a queryService whose value is a SearchService. A SearchService is a subclass of InternetService (which is defined elsewhere). A SearchQuery has some number of result properties (whose value is SearchResult). Each SearchResult has a title (value is a rdfs:Literal), a rating and of course, the page itself.

The modularity of RDF allows other vocabularies to be combined with simple schemas such as this to characterize more fully the properties of networked resources. For example, Dublin Core or a library-based classification vocabulary might be used to describe the subject coverage or collections-level properties for each SearchService, while an independently managed "search protocols" vocabulary could be used to describe connection details for (say) LDAP, WHOIS++ or Z39.50 search interfaces offered by the service. By allowing the creation of statements which draw upon specialized schemas from various domains, RDF makes it possible for diverse communities of expertise to contribute to a decentralized web of machine-readable vocabularies.
<rdf:RDF xml:lang="en"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <rdfs:Class rdf:ID="SearchQuery">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
  </rdfs:Class>
  <rdfs:Class rdf:ID="SearchResult">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
  </rdfs:Class>
  <rdfs:Class rdf:ID="SearchService">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/03/example/classes#InternetService"/>
  </rdfs:Class>
  <rdf:Property ID="queryString">
    <rdfs:domain rdf:resource="#SearchQuery"/>
    <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
  </rdf:Property>
  <rdf:Property ID="queryService">
    <rdfs:domain rdf:resource="#SearchQuery"/>
    <rdfs:range rdf:resource="#SearchService"/>
  </rdf:Property>
  <rdf:Property ID="result">
    <rdfs:domain rdf:resource="#SearchQuery"/>
    <rdfs:range rdf:resource="#SearchResult"/>
  </rdf:Property>
  <rdf:Property ID="queryResultPage">
    <rdfs:domain rdf:resource="#SearchResult"/>
    <rdfs:range rdf:resource="http://www.w3.org/2000/03/example/classes#WebPage"/>
  </rdf:Property>
  <rdf:Property ID="queryResultTitle">
    <rdfs:domain rdf:resource="#SearchResult"/>
    <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
  </rdf:Property>
  <rdf:Property ID="queryResultRating">
    <rdfs:domain rdf:resource="#SearchResult"/>
    <rdfs:range rdf:resource="http://www.w3.org/2000/03/example/classes#FloatZeroToOne"/>
  </rdf:Property>
</rdf:RDF>
8. References

8.1. Normative References

http://www.w3.org/TR/1999/REC-rdf-syntax-19990222

/XMLNS] Namespaces in XML; W3C Recommendation, 14 January 1999
http://www.w3.org/TR/1999/REC-xml-names-19990114

8.2. Informational References


http://www.w3.org/DesignIssues/RDFnot.html

[CycL] CycL: The CYC Representation Language

[KIF] Knowledge Interchange Format (KIF)

http://cs-tr.cs.cornell.edu/Dienst/UI/2.0/Describe/ncstrl.cornell/TR96-1593

http://www.w3.org/TR/1998/NOTE-webarch-extlang-19980210

[PICS] Platform for Internet Content Selection;
http://www.w3.org/PICS/

http://www.w3.org/TR/2000/NOTE-rdf-pics-20000327

[DC] Dublin Core Metadata Initiative
http://purl.org/dc/

http://www.w3.org/TR/1999/NOTE-schema-arch-19991007

[UML] Unified Modeling Language (UML)
http://www.rational.com/uml/resources

/XML] Extensible Markup Language (XML) 1.0, W3C Recommendation, 10-February-1988, Section 3.2 Element Type Declarations
http://www.w3.org/TR/1998/REC-xml-19980210.html#elemdecls

8. Acknowledgements

Note: This document was prepared and approved for publication by the W3C RDF Schema Working Group (WG). WG approval of this document does not necessarily imply that all WG members voted for its approval.

David Singer of IBM was the chair of the group throughout most of the development of this specification; we thank David for his efforts and thank IBM for supporting him and us in this endeavor. Particular thanks are also due to Andrew Layman for his editorial work on earlier versions of this specification.

The working group membership has included:

Nick Arnett (Verity), Dan Brickley (ILRT / University of Bristol), Walter Chang (Adobe), Sailesh Chutani (Oracle), Ron Daniel (DATAFUSION), Charles Frankston (Microsoft), Joe Lapp (webMethods Inc.), Patrick Gannon (CommerceNet), RV Guha (Epinions, previously of Netscape Communications), Tom Hill (Apple Computer), Renato Iannella (DSTC), Sandeep Jain (Oracle), Kevin Jones, (InterMind), Emiko Kezuka (Digital Vision Laboratories), Ora Lassila (Nokia Research Center), Andrew Layman (Microsoft), John McCarthy (Lawrence Berkeley National Laboratory), Michael Mealling (Network Solutions), Norbert Mikula (DataChannel), Eric Miller (OCLC), Frank Olken (Lawrence Berkeley National Laboratory), Sri Raghavan (Digital/Compaq), Lisa Rein (webMethods Inc.), Tsuyoshi Sakata (Digital Vision Laboratories), Leon Shklar (Pencom Web Works), David Singer (IBM), Wei (William) Song (SISU), Neel Sundaresan (IBM), Ralph Swick (W3C), Naohiko Uramoto (IBM), Charles Wicksteed (Reuters Ltd.), Misha Wolf (Reuters Ltd.)

Not all of the people listed above have been members throughout the entire duration of the working group, but all have contributed to the evolution of this document.

Appendix A: XML Serialization

An RDF specification of the core RDF Schema model is given here in RDF/XML serialization syntax. Please note that the namespace URI for the RDF Schema Specification will change in future versions of this specification if the schema changes. This RDF schema includes annotations describing RDF resources defined formally in the RDF Model and Syntax specification, as well as definitions for new resources belonging to the RDF Schema namespace.

Note that there are some constraints (such as those given in 2.3.2 above) on certain RDF Schema resources which are themselves not fully expressible using the RDF Schema specification. For example, the RDF below does not tell us that subClassOf arcs should not (to use terminology from the nodes and arcs representation) form loops in any RDF model.
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
 <rdfs:Class rdf:ID="Resource">
   <rdfs:label xml:lang="en">Resource</rdfs:label>
   <rdfs:label xml:lang="fr">Ressource</rdfs:label>
   <rdfs:comment>The most general class</rdfs:comment>
 </rdfs:Class>
 <rdf:Property about="http://www.w3.org/1999/02/22-rdf-syntax-ns#type">
   <rdfs:label xml:lang="en">type</rdfs:label>
   <rdfs:label xml:lang="fr">type</rdfs:label>
   <rdfs:comment>Indicates membership of a class</rdfs:comment>
   <rdfs:range rdf:resource="#Class"/>
 </rdf:Property>
 <rdf:Property ID="comment">
   <rdfs:label xml:lang="en">comment</rdfs:label>
   <rdfs:label xml:lang="fr">commentaire</rdfs:label>
   <rdfs:domain rdf:resource="#Resource"/>
   <rdfs:comment>Use this for descriptions</rdfs:comment>
   <rdfs:range rdf:resource="#Literal"/>
 </rdf:Property>
 <rdf:Property ID="label">
   <rdf:type resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
   <rdfs:label xml:lang="en">label</rdfs:label>
   <rdfs:label xml:lang="fr">label</rdfs:label>
   <rdfs:domain rdf:resource="#Resource"/>
   <rdfs:comment>Provides a human-readable version of a resource name.</rdfs:comment>
   <rdfs:range rdf:resource="#Literal"/>
 </rdf:Property>
 <rdfs:Class rdf:ID="Class">
   <rdfs:label xml:lang="en">Class</rdfs:label>
   <rdfs:label xml:lang="fr">Classe</rdfs:label>
   <rdfs:comment>The concept of Class</rdfs:comment>
   <rdfs:subClassOf rdf:resource="#Resource"/>
 </rdfs:Class>
 <rdf:Property ID="subClassOf">
   <rdfs:label xml:lang="en">subClassOf</rdfs:label>
   <rdfs:label xml:lang="fr">sousClasseDe</rdfs:label>
   <rdfs:comment>Indicates membership of a class</rdfs:comment>
   <rdfs:range rdf:resource="#Class"/>
   <rdfs:domain rdf:resource="#Class"/>
 </rdf:Property>
 <rdf:Property ID="subPropertyOf">
   <rdfs:label xml:lang="en">subPropertyOf</rdfs:label>
   <rdfs:label xml:lang="fr">sousPropriétéDe</rdfs:label>
   <rdfs:comment>Indicates specialization of properties</rdfs:comment>
   <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
   <rdfs:domain rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
 </rdf:Property>
</rdf:RDF>
<rdf:Property ID="seeAlso">
  <rdfs:label xml:lang="en">seeAlso</rdfs:label>
  <rdfs:label xml:lang="fr">voirAussi</rdfs:label>
  <rdfs:comment>Indicates a resource that provides information about the subject resource.</rdfs:comment>
  <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
  <rdfs:domain rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
</rdf:Property>

<rdfs:Property ID="isDefinedBy">
  <rdfs:type resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  <rdfs:subPropertyOf rdf:resource="#seeAlso"/>
  <rdfs:label xml:lang="en">isDefinedBy</rdfs:label>
  <rdfs:label xml:lang="fr">esDéfiniPar</rdfs:label>
  <rdfs:comment>Indicates a resource containing and defining the subject resource.</rdfs:comment>
  <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
  <rdfs:domain rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
</rdf:Property>

<rdfs:Class rdf:ID="ConstraintResource">
  <rdfs:label xml:lang="en">ConstraintResource</rdfs:label>
  <rdfs:label xml:lang="fr">RessourceContrainte</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Resource"/>
  <rdfs:comment>Resources used to express RDF Schema constraints.</rdfs:comment>
</rdfs:Class>

<rdfs:Class rdf:ID="ConstraintProperty">
  <rdfs:label xml:lang="en">ConstraintProperty</rdfs:label>
  <rdfs:label xml:lang="fr">PropriétéContrainte</rdfs:label>
  <rdfs:subClassOf rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  <rdfs:subClassOf rdf:resource="#ConstraintResource"/>
  <rdfs:comment>Properties used to express RDF Schema constraints.</rdfs:comment>
</rdfs:Class>

<rdfs:ConstraintProperty rdf:ID="domain">
  <rdfs:label xml:lang="en">domain</rdfs:label>
  <rdfs:label xml:lang="fr">domaine</rdfs:label>
  <rdfs:comment>This is how we associate a class with properties that its instances can have</rdfs:comment>
</rdfs:ConstraintProperty>

<rdfs:ConstraintProperty rdf:ID="range">
  <rdfs:label xml:lang="en">range</rdfs:label>
  <rdfs:label xml:lang="fr">étendue</rdfs:label>
  <rdfs:comment>Properties that can be used in a schema to provide constraints</rdfs:comment>
  <rdfs:range rdf:resource="#Class"/>
  <rdfs:domain rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
</rdfs:ConstraintProperty>

<rdfs:Class rdf:about="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property">
  <rdfs:label xml:lang="en">Property</rdfs:label>
  <rdfs:label xml:lang="fr">Propriété</rdfs:label>
  <rdfs:comment>The concept of a property.</rdfs:comment>
  <rdfs:subClassOf rdf:resource="#Resource"/>
</rdfs:Class>
<rdfs:Class rdf:ID="Literal">
  <rdfs:label xml:lang="en">Literal</rdfs:label>
  <rdfs:label xml:lang="fr">Littéral</rdfs:label>
  <rdf:type resource="#Class"/>
  <rdfs:comment>This represents the set of atomic values, eg. textual strings.</rdfs:comment>
</rdfs:Class>

<rdfs:Class rdf:about="http://www.w3.org/1999/02/22-rdf-syntax-ns#Statement">
  <rdfs:label xml:lang="en">Statement</rdfs:label>
  <rdfs:label xml:lang="fr">Déclaration</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Resource"/>
  <rdfs:comment>This represents the set of reified statements.</rdfs:comment>
</rdfs:Class>

<rdf:Property about="http://www.w3.org/1999/02/22-rdf-syntax-ns#subject">
  <rdfs:label xml:lang="en">subject</rdfs:label>
  <rdfs:label xml:lang="fr">sujet</rdfs:label>
  <rdfs:domain rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Statement"/>
  <rdfs:range rdf:resource="#Resource"/>
</rdf:Property>

<rdf:Property about="http://www.w3.org/1999/02/22-rdf-syntax-ns#predicate">
  <rdfs:label xml:lang="en">predicate</rdfs:label>
  <rdfs:label xml:lang="fr">prédicat</rdfs:label>
  <rdf:type resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  <rdfs:domain rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Statement"/>
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
</rdf:Property>

<rdf:Property about="http://www.w3.org/1999/02/22-rdf-syntax-ns#object">
  <rdfs:label xml:lang="en">object</rdfs:label>
  <rdfs:label xml:lang="fr">objet</rdfs:label>
  <rdfs:domain rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Statement"/>
</rdf:Property>

<rdfs:Class rdf:ID="Container">
  <rdfs:label xml:lang="en">Container</rdfs:label>
  <rdfs:label xml:lang="fr">Enveloppe</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Resource"/>
  <rdfs:comment>This represents the set Containers.</rdfs:comment>
</rdfs:Class>

<rdfs:Class rdf:about="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag">
  <rdfs:label xml:lang="en">Bag</rdfs:label>
  <rdfs:label xml:lang="fr">Ensemble</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Container"/>
</rdfs:Class>

<rdfs:Class rdf:about="http://www.w3.org/1999/02/22-rdf-syntax-ns#Seq">
  <rdfs:label xml:lang="en">Sequence</rdfs:label>
  <rdfs:label xml:lang="fr">Séquence</rdfs:label>
  <rdfs:subClassOf rdf:resource="#Container"/>
</rdfs:Class>

<rdfs:Class rdf:about="http://www.w3.org/1999/02/22-rdf-syntax-ns#Alt">
  <rdfs:label xml:lang="en">Alt</rdfs:label>
  <rdfs:label xml:lang="fr">Choix</rdfs:label>
</rdfs:Class>
Appendix B. About W3C Documents

To promote confidence and stability, W3C has instituted the following publication policies:

1. Each technical report is a stable published document that has a unique identifier (URI).
2. Each stable published document will always be available, unchanged, at that URI. Retrieving (e.g., by bookmarking) the resource at that URI will always return the same content.
3. A specific revision of a document is generally one of a series of related documents (e.g., from Working Draft to Recommendation). Each series has a unique identifier (URI) that when followed, will return the latest stable published document available in the series.
4. Each document includes a "Status" section that describes the document's publication context on the date of publication. Note that since W3C does not change stable published documents, the status section of a published document cannot be changed, even if the document becomes obsolete at a later date.
5. Each document in the series includes links to up-to-date status information for the series.

Comments

$Date: 2000/03/27 19:14:02 $