



## RDF Schema und Dublin Core

RDF [RDFM&S] ermöglicht die gleichzeitige Nutzung unterschiedlicher Vokabulare für die Beschreibung von Metadaten.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/DC/documents/rec-dces-19990702.htm"
  xmlns:vCard="http://imc.org/vCard/3.0#">
  <rdf:Description ID="Document">
    <dc:creator>
      <rdf:Description>
        <vCard:FN>Hartmut Polzer</vCard:FN>
        <vCard:EMAIL>hartmut@gmx.de</vCard:EMAIL>
      </rdf:Description>
    </dc:creator>
  </rdf:Description>
</rdf:RDF>
```

Mit Hilfe von RDF Schema [RDFS] lassen sich

- Vokabulare in wohldefinierter Weise spezifizieren und
- Relationen zwischen Vokabularen definieren.

Das RDF Schema des Dublin Core wird im Folgenden ausführlich vorgestellt werden und es werden mögliche Erweiterungen bezüglich der Darstellung von Qualified Dublin Core thematisiert. Anhand dieser konkreten Beispiele werden die grundlegenden Konzeptionen von RDF Schema erläutert.

### Ein RDF Schema für Dublin Core

Der folgende Ausschnitt zeigt einen Teil des RDF Schema für Dublin Core:

```
<? xml version='1.0'?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/TR/REC-rdf-syntax#"
  xmlns:rdfs="http://www.w3.org/TR/WD-rdf-schema#"
  xmlns:dc="">
  <rdf:Description ID="title">
    <rdf:type rdf:resource="http://www.w3.org/TR/REC-rdf-syntax#Property"/>
    <rdfs:label>Title</rdfs:label>
    <rdfs:comment>The name given to the resource, usually by the Creator
    or Publisher.</rdfs:comment>
    <rdfs:isDefinedBy = ""/>
  </rdf:Description>

  <rdf:Description ID="creator">
    <rdf:type rdf:resource="http://www.w3.org/TR/REC-rdf-syntax#Property"/>
    <rdfs:label>Author/Creator</rdfs:label>
    <rdfs:comment>The person or organization primarily responsible for
    creating the intellectual content of the resource. For example,
```

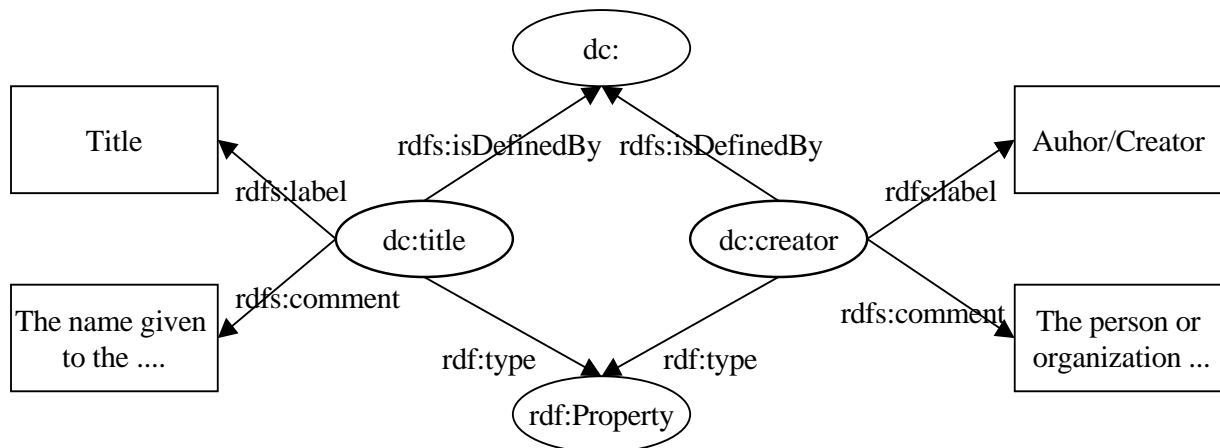
```

    authors in the case of written documents, artists, photographers, or
    illustrators in the case of visual resources.</rdfs:comment>
    <rdfs:isDefinedBy = ""/>
  </rdf:Description>
...
</rdf:RDF>

```

(Hierbei bezeichnet "" die URI des Dokumentes, in dem dieses RDF enthalten ist. Im obigen Fall ist das die URL <http://purl.org/DC/documents/rec-dces-19990702.htm>.)

Ein RDF Schema ist ein RDF Dokument, in dem spezielle Eigenschaften (aus dem `rdfs` Namespace) benutzt werden! Syntaktisch unterscheidet es sich damit überhaupt nicht von den RDF Dokumenten, die bislang vorgestellt wurden. Das RDF Graphenmodell zur obigen XML Repräsentation sieht wie folgt aus:



(Der Übersichtlichkeit halber sind die URIs der Resources mit Hilfe von Präfixen für die jeweiligen Namespaces abgekürzt.)

Bei der Spezifikation eines RDF Schema werden im wesentlichen Aussagen über Resources gemacht, die vom Typ `rdf:Property` und `rdfs:Class` sind. Die so definierten Properties können später zur (semantischen) Beschreibung von Daten verwendet werden. Welche speziellen Properties für die Spezifikation von neuen Vokabularen zur Verfügung stehen, ist im `rdfs` Namespace definiert. In dem RDF Schema des Dublin Core werden folgende Properties aus dem `rdfs` Namespace für die Beschreibung der Dublin Core Elemente benutzt:

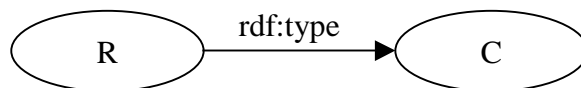
- `rdfs:label`. Ein Name bzw. eine Bezeichnung für die Resource.
- `rdfs:comment`. Eine menschenlesbare Beschreibung der Resource.
- `rdfs:isDefinedBy`. Ein Pfeil auf eine Resource, die die zu beschreibende Resource definiert.

Für eine vollständige Auflistung der im `rdfs` Namespace definierten Properties sei auf die Spezifikation [RDFS] verwiesen. Im nächsten Abschnitt werden die grundlegenden Konzepte der RDF Schema Spezifikation vorgestellt.

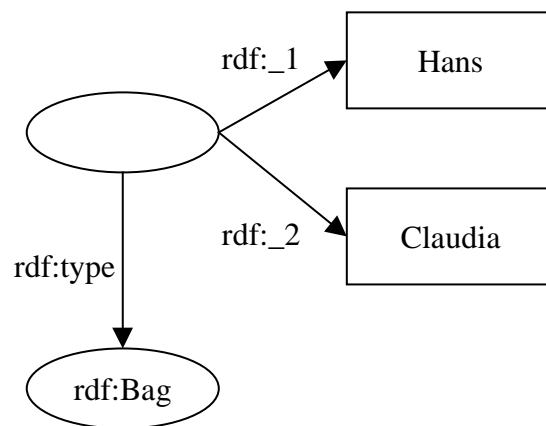
## Klassen und Eigenschaften

### Klassen

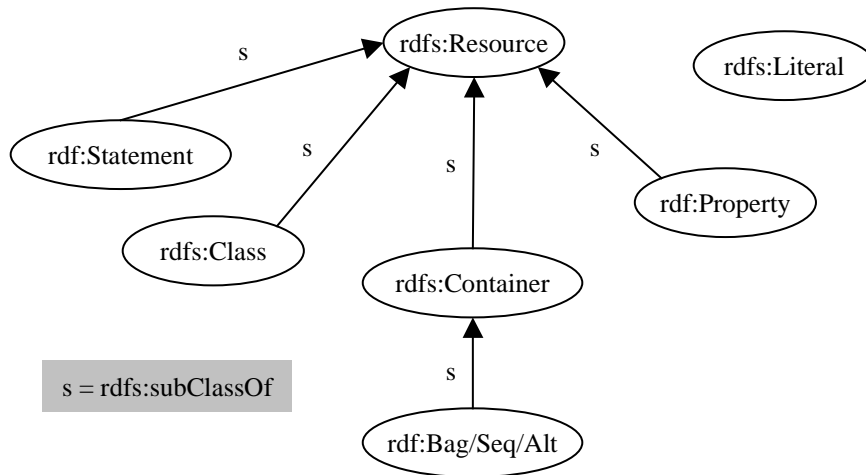
RDF Schema unterstützt die Definition von Klassen analog zu objektorientierten Programmiersprachen. Eine Klasse beschreibt eine Menge von Resources, die bestimmte Eigenschaften haben. Ist eine Resource R Element (oder Instanz) einer Klasse C, so wird dies durch den RDF Graphen



repräsentiert. Dabei ist die Klasse C selbst wieder durch eine Resource gegeben. Einige Klassen sind schon in der RDF Spezifikation definiert. Zum Beispiel ist ein Bag definiert als eine Resource, von der aus  $n$  Pfeile `rdf:_1`, ..., `rdf:_n` ( $n \geq 0$ ) starten (natürlich können auch noch weitere Prädikate an dieser Resource starten):

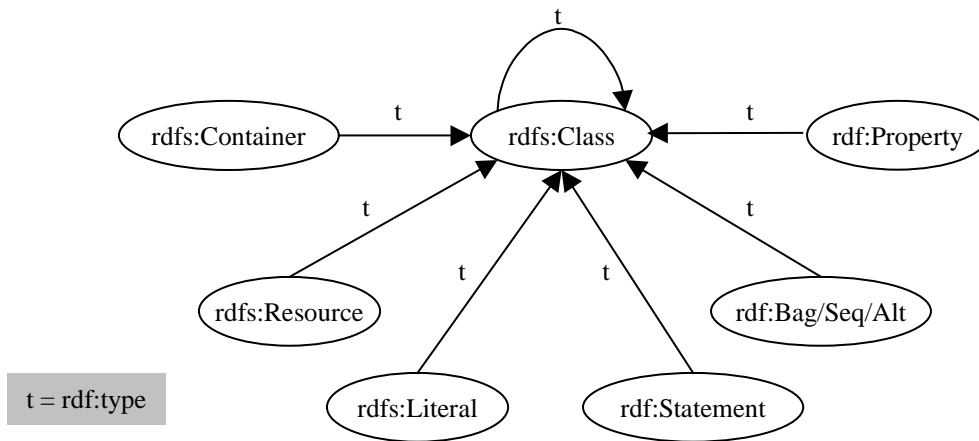


Eine Klasse A heisst Unterklasse einer Klasse B, wenn die Menge der durch A definierten Resources eine Teilmenge der Menge der durch B definierten Resources ist. Diese Relation zwischen Klassen wird durch die Property `rdfs:subClassOf` ausgedrückt. Der folgende RDF Graph zeigt einen Ausschnitt dieser Relation für Klassen aus [RDFM&S] und [RDFS]:

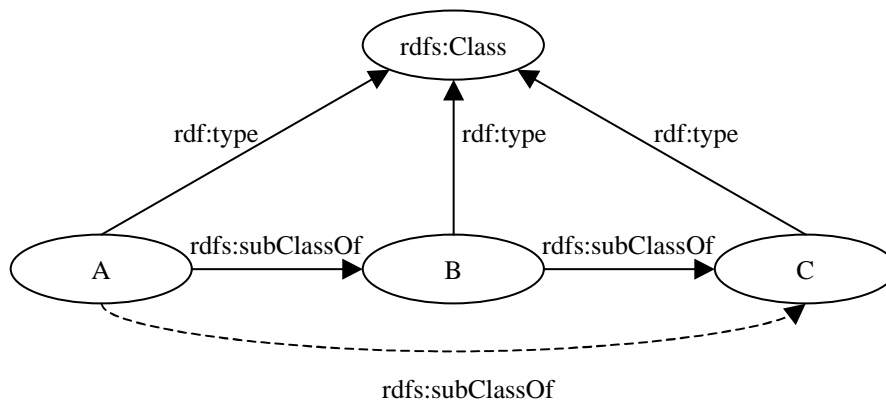


- **rdfs:Literals.** Die Klasse der "Literals", die in [RDFM&S] definiert wurde. Da über Literals keine Aussagen gemacht werden dürfen, kann die Klasse der Literals keine Unterklasse einer der anderen Klassen sein. (Sonst wäre sie auch eine Unterklasse der Resources. Widerspruch.)
- **rdfs:Resource.** Die Klasse aller Objekte, über die mit Hilfe von RDF Aussagen gemacht werden [RDFM&S]. Die Klasse der Resources ist Oberklasse von allen anderen Klassen, ausser der Klasse der Literals. Das heisst jedes Statement, jede Klasse, jeder Container, jede Property ist eine Resource!
- **rdf:Statement.** Dies ist die Klasse der Statements, die in [RDFM&S] definiert wurde. Eine Resource aus der Klasse `rdf:Statement` besitzt jeweils genau eine der folgenden drei Properties: `rdf:predicate`, `rdf:object` und `rdf:subject`.
- **rdfs:class.** Das Konzept von Klassen entspricht dem aus objektorientierten Programmiersprachen bekanntem Paradigma. Eine Resource aus der Klasse `rdfs:class` ist dadurch charakterisiert, dass sie eine Property `rdf:type` besitzt, deren Wert `rdfs:Class` ist.
- **rdf:Property.** Die Klasse der Properties beschreibt diejenigen Resources, die zur Beschreibung von Resources benutzt werden.
- **rdfs:Container, rdfs:Bag/Seq/Alt.** Die Klasse `rdfs:Container` ist die Oberklasse aller Container Objekte. Die unterschiedliche Semantik der drei Unterklassen `Bag`, `Seq` oder `Alt` ist in der RDF Spezifikation beschrieben.

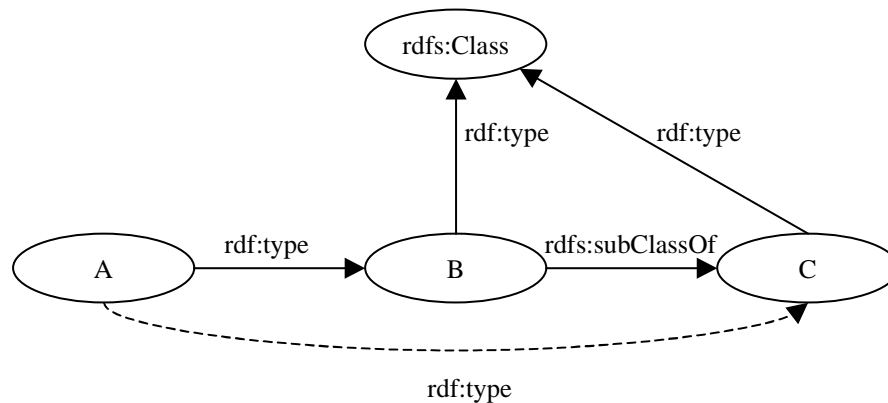
Bislang ist die durch die Property `rdfs:subClassOf` ('Teilmenge von') definierte Relation zwischen Klassen betrachtet worden. In der folgenden Grafik wird die Relation `rdf:type` ('ist Element von') repräsentiert:



Das bedeutet: Jede Klasse ist Element der Klasse aller RDF Klassen! Insbesondere ist die Klasse aller Klassen eine Klasse. Konstruktionen dieser Art sind in der Mathematik wohlbekannt, auch wenn sie nicht immer einfach zu interpretieren sind. Man sollte sie als Axiome auffassen. Wichtiger sind die Schlußfolgerungen, die man aus ihnen ziehen kann. Zum Beispiel wird die Property `rdfs:subClassOf` als transitive Relation definiert, d.h. ist A Unterklasse von B und B Unterklasse von C, so ist A Unterklasse von C:



Auf diese Weise gewinnt man neue Relationen und somit neue Information. Ein weiteres Axiom ist das folgende: Ist A Element der Klasse B und B Unterklasse von C, so ist auch A Element aus C:



### Beispiel 1: Klassen und Qualified Dublin Core

Die Dublin Core Initiative hat eine Liste von Qualifiern für die 15 Hauptelemente verabschiedet. Die Qualifier werden grob in zwei Klassen aufgeteilt: in die Element Refinements und die Encoding Schemes. Die Element Refinements beschreiben semantische Verfeinerungen der Hauptelemente, während die Encoding Schemes Vokabulare für die Beschreibung der Elementwerte zur Verfügung stellen (z.B. Klassifikationssysteme). Um diese Qualifier in RDF nutzen zu können, müssen sie (analog zu den Hauptelementen) innerhalb eines RDF Schema definiert sein. Im Folgenden wird an dem Beispiel von `dc:subject` gezeigt, wie eine solche Definition aussehen könnte. Dazu betrachte man die ursprünglich vorgesehenen Qualifier von `dc:subject`:

- Element Refinement: `classification`
- Encoding Schemes: LCSH, MESH, DDC, LCC, UDC.

Es darf angezweifelt werden, ob `classification` wirklich eine semantische Verfeinerung von `dc:subject` ist. Schliesslich handelt es sich ja immer noch um das 'subject' des zu beschreibenden Objektes, nur die Art der Beschreibung hat sich geändert. Während vorher Schlüsselwörter verwendet wurden, wird nun ein Klassifikationssystem für die Beschreibung von 'subject' benutzt. Somit hat sich nur der Typ der Beschreibung geändert, nicht die Beziehung zu dem zu beschreibenden Objekt.

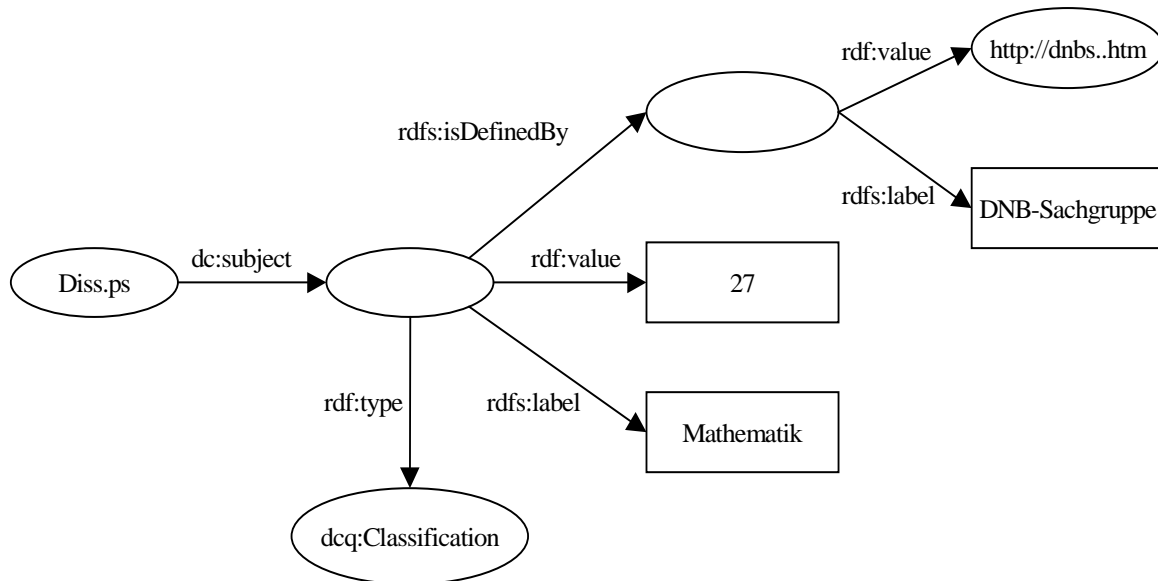
Wahrscheinlich ist auf Grund dieser Problematik `classification` als Qualifier von `dc:subject` abgelehnt worden. Im Folgenden wird gezeigt, wie sich dieses Problem mit Hilfe von RDF elegant lösen lässt.

Es liegt nahe, eine Klasse für den Qualifier `classification` zu definieren. Diese Klasse sollte folgende Werte repräsentieren können:

- Name der verwendeten Klassifikation

- URL der Klassifikation (falls vorhanden)
- (Aktueller) Wert
- Menschenverstehbare Formulierung des Wertes

Nehmen wir an, wir wollen eine Dissertation mit Hilfe der DNB-Sachgruppe klassifizieren. Der entsprechende RDF Graph könnte folgendermaßen aussehen:



Dass eine Resource vom Typ `Classification` unabhängig von `dc:subject` ist (und somit keine semantische Verfeinerung ist), kann man sich folgendermaßen verdeutlichen. Angenommen, in dem (fiktiven) Namespace `ns` gibt es eine Property `ns:interests`, die die Interessensgebiete eines Wissenschaftlers beschreiben soll. Ersetzt man nun im obigen Graphen die Dissertation `Diss.ps` durch eine Resource, die einen Wissenschaftler repräsentiert und die Property `dc:subject` durch `ns:interests`, so ergibt sich ein semantisch sinnvoller RDF Graph.

Der folgende RDF Ausschnitt skizziert eine mögliche Definition der Klasse `Classification` innerhalb des `dcq` (Dublin Core Qualifier) Namespaces:

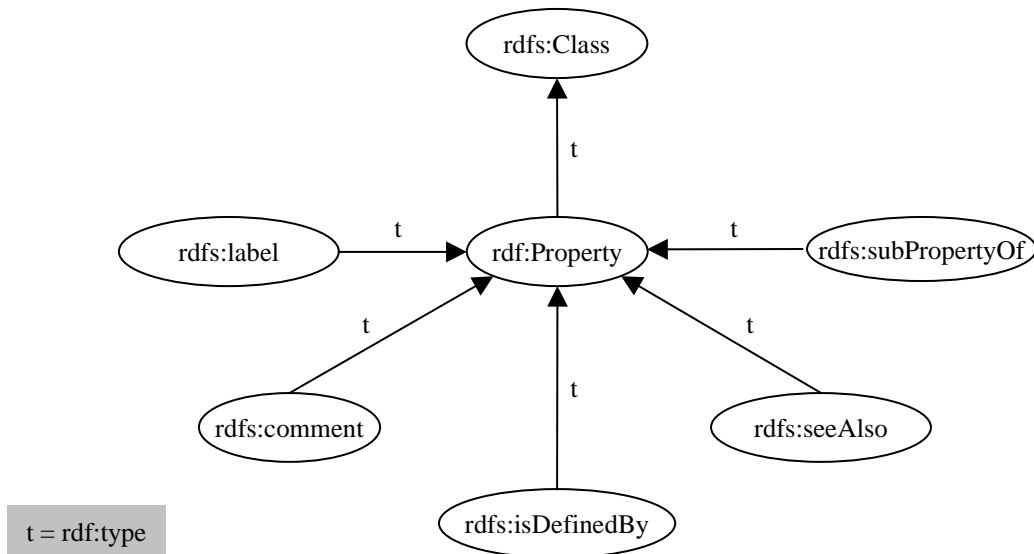
```

<rdf:Description ID="Classification">
  <rdf:type rdf:resource="http://www.w3.org/TR/2000/CR-rdf-schema-20000327/Class"/>
  <rdfs:label>Classification </rdfs:label>
  <rdfs:comment>Eine Resource vom Typ Klassifikation ist definiert durch ...</rdfs:comment>
  <rdfs:isDefinedBy = ""/>
</rdf:Description>

```

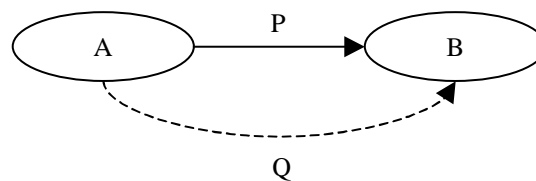
## Properties

Neben der Definition von Klassen ermöglicht RDF Schema natürlich die Definition von neuen Properties. Die folgende Grafik zeigt die wichtigsten Properties, die zur Definition



von neuen Vokabularen benutzt werden können:

Mit `rdfs:isSubPropertyOf` wird (analog zu `rdfs:subClassOf` bei Klassen) eine transitive Relation definiert. Ist eine Property P `rdfs:subPropertyOf` einer Property Q, folgt aus dem Statement (A,P,B) das Statement (A,Q,B):



### Beispiel 2: Properties und Qualified Dublin Core

Mit Hilfe der Relation `rdfs:subPropertyOf` lässt sich genau die semantische Verfeinerung realisieren, die bei Qualifiern vom Typ Element Refinement benötigt wird. Man betrachte z.B. den Qualifier `created` vom Hauptelement `dc:date`. Der Qualifier `created` verfeinert die Semantik des Elementes `dc:date` in Bezug auf das zu beschreibende Dokument und kann deshalb in diesem Fall nicht durch eine neue Klasse angemessen repräsentiert werden. Das folgende RDF Fragment definiert die neue Property `dcq:created` als Subproperty von `dc:date`.



## Referenzen

[RDFS] Resource Description Framework (RDF) Schema Specification 1.0  
<http://www.w3.org/TR/2000/CR-rdf-schema-20000327/>

[RDFM&S] Resource Description Framework (RDF) Model and Syntax Specification  
<http://www.w3.org/TR/REC-rdf-syntax>