Semantic Web Engineering

Gerald Reif

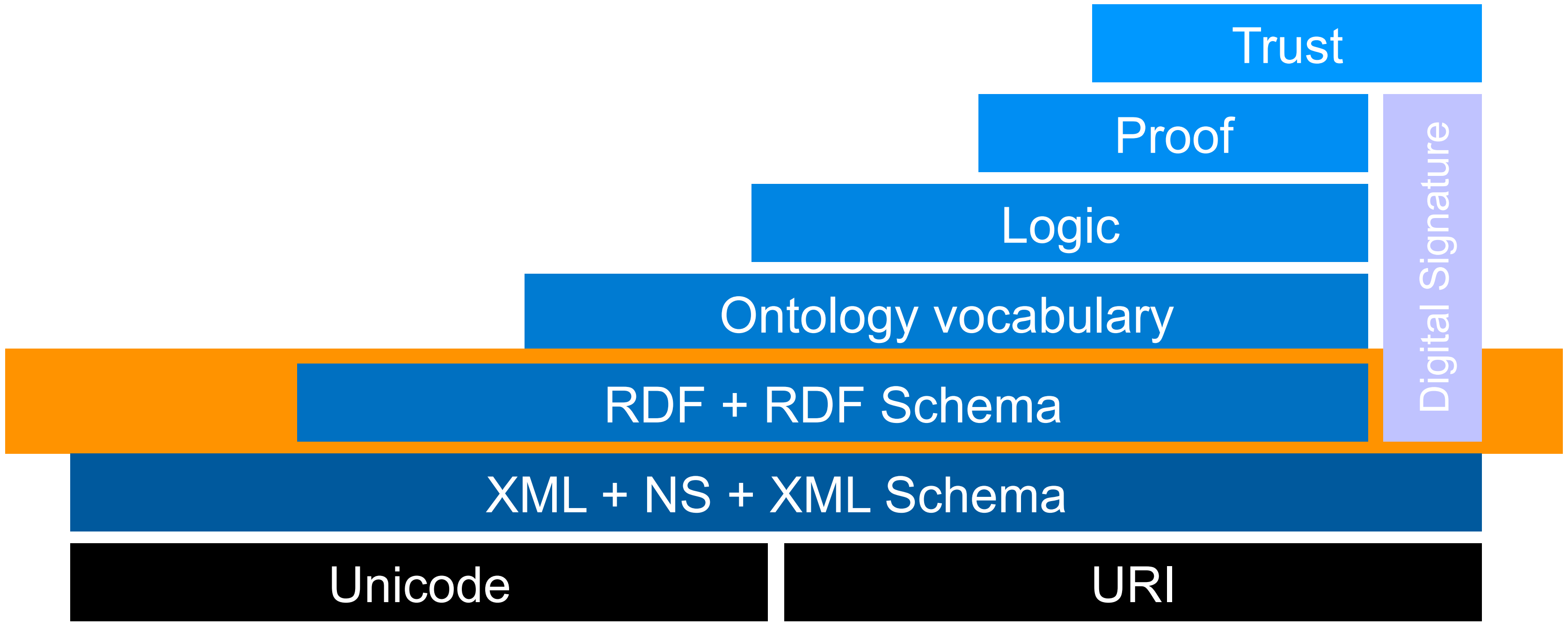
reif@ifi.unizh.ch

Fr. 10:15-12:00, Room 2.A.10

University of Zurich
Department of Informatics

s.e.a.l.
software evolution & architecture lab

RDF Schema


Trust
Proof
Logic
Ontology vocabulary
RDF + RDF Schema
XML + NS + XML Schema
Unicode
URI
Digital Signature

Defining RDF Vocabulary: RDF Schema

- RDF provides a way to express simple statements about resources, using named properties and values.
- RDF user communities also need the ability to define the vocabularies (terms, concepts) they intend to use in those statements, specifically,
  - to indicate that they are describing specific kinds or classes of resources,
  - and will use specific properties in describing those resources.
- For example, to describe classes such as `exterms:Tent`, and use properties such as `exterms:model`, `exterms:weightInKg`, and `exterms:packedSize` to describe them.

RDF Schema 1/2

- RDF itself provides no means for defining such application-specific classes and properties.
- Instead, such classes and properties are described as an RDF vocabulary, using extensions to RDF provided by the RDF Vocabulary Description Language 1.0: RDF Schema.
  - http://www.w3.org/TR/rdf-schema/
- RDF Schema does not provide a vocabulary of application-specific classes like `exterms:Tent` and properties like `exterms:weightInKg`.
- Instead, it provides the facilities needed to describe such classes and properties.
- In other words, RDF Schema provides a type system for RDF.
- This type system is also called ontology.

RDF Schema 2/2

- The RDF Schema facilities are themselves provided in the form of an RDF vocabulary;
  - that is, as a specialized set of predefined RDF resources with their own special meanings.
- The resources in the RDF Schema vocabulary have URIrefs with the prefix `http://www.w3.org/2000/01/rdf-schema#` (conventionally associated with the QName prefix `rdfs:`).
- Vocabulary descriptions (schemas) written in the RDF Schema language are legal RDF graphs.

Defining Classes 1/2

- A basic step in any kind of description process is identifying the various kinds of things to be described.
- RDF Schema refers to these "kinds of things" as classes.
- A class in RDF Schema corresponds to the generic concept of a Type or Category, somewhat like the notion of a class in object-oriented programming languages such as Java.
- RDF classes can be used to represent almost any category of thing, such as Web pages, people, document types, databases or abstract concepts.
- Classes are described using the RDF Schema resources `rdfs:Class` and `rdfs:Resource`, and the properties `rdf:type` and `rdfs:subClassOf`.
- Resources that belong to a class are called its instances.

Defining Classes 2/2

- In RDF Schema, a class is any resource having an `rdf:type` property whose value is the resource `rdfs:Class`.

```
ex:MotorVehicle   rdf:type   rdfs:Class .
```

- The property `rdf:type` is used to indicate that a resource is an instance of a class.

```
exthings:companyCar   rdf:type  ex:MotorVehicle .
```

- A subclass is defined using the `rdfs:subClassOf` property.

```
ex:Van   rdf:type   rdfs:Class .
ex:Van   rdfs:subClassOf   ex:MotorVehicle .
```

- RDF Schema defines all classes as subclasses of class `rdfs:Resource` (since the instances belonging to all classes are resources).

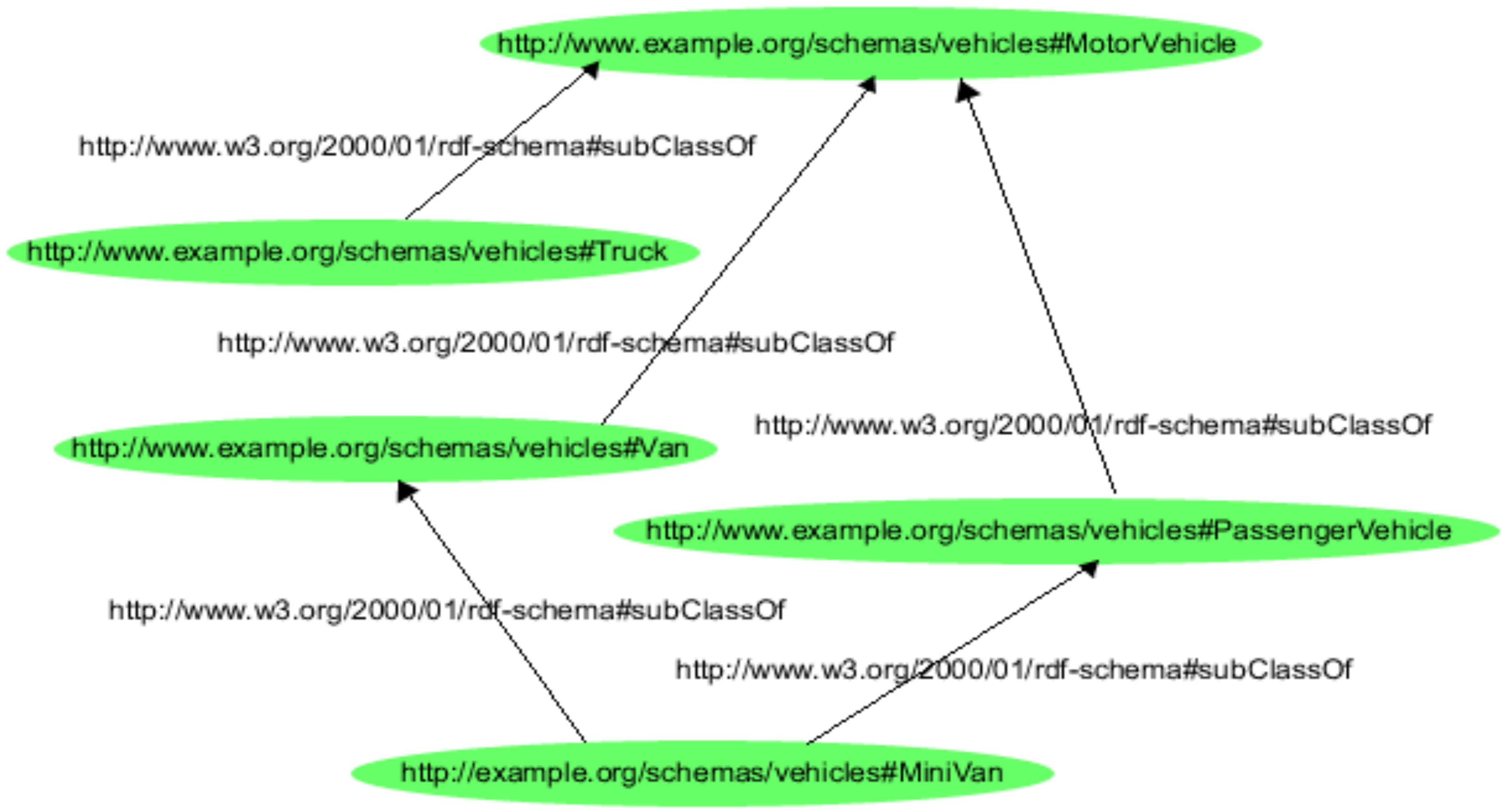
Example Class Definitions

```
ex:MotorVehicle      rdf:type         rdfs:Class .
ex:PassengerVehicle  rdf:type         rdfs:Class .
ex:Van               rdf:type         rdfs:Class .
ex:Truck             rdf:type         rdfs:Class .
ex:MiniVan           rdf:type         rdfs:Class .

ex:PassengerVehicle  rdfs:subClassOf  ex:MotorVehicle .
ex:Van               rdfs:subClassOf  ex:MotorVehicle .
ex:Truck             rdfs:subClassOf  ex:MotorVehicle .

ex:MiniVan           rdfs:subClassOf  ex:Van .
ex:MiniVan           rdfs:subClassOf  ex:PassengerVehicle .
```

Class Definition as RDF graph


http://www.example.org/schemas/vehicles#MotorVehicle
http://www.w3.org/2000/01/rdf-schema#subClassOf
http://www.example.org/schemas/vehicles#Truck
http://www.w3.org/2000/01/rdf-schema#subClassOf
http://www.w3.org/2000/01/rdf-schema#subClassOf
http://www.example.org/schemas/vehicles#Van
http://www.example.org/schemas/vehicles#PassengerVehicle
http://www.w3.org/2000/01/rdf-schema#subClassOf
http://www.w3.org/2000/01/rdf-schema#subClassOf
http://example.org/schemas/vehicles#MiniVan

Class Definition in RDF/XML 1/2

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
      xml:base="http://example.org/schemas/vehicles">
   <rdf:Description rdf:ID="MotorVehicle">
      <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
   </rdf:Description>
   <rdf:Description rdf:ID="PassengerVehicle">
      <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
      <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
   </rdf:Description>
   <rdf:Description rdf:ID="Van">
      <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
      <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
   </rdf:Description>
   <rdf:Description rdf:ID="MiniVan">
      <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
      <rdfs:subClassOf rdf:resource="#Van"/>
      <rdfs:subClassOf rdf:resource="#PassengerVehicle"/>
   </rdf:Description>
</rdf:RDF>
```

Class Definition in RDF/XML 2/2

```
<rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
      xml:base="http://example.org/schemas/vehicles">
   <rdfs:Class rdf:ID="MotorVehicle"/>
   <rdfs:Class rdf:ID="PassengerVehicle">
      <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
   </rdfs:Class>
   <rdfs:Class rdf:ID="Truck">
      <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
   </rdfs:Class>
   <rdfs:Class rdf:ID="Van">
      <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
   </rdfs:Class>
   <rdfs:Class rdf:ID="MiniVan">
      <rdfs:subClassOf rdf:resource="#Van"/>
      <rdfs:subClassOf rdf:resource="#PassengerVehicle"/>
   </rdfs:Class>
</rdf:RDF>
```

Describing Properties

- We also need to be able to describe specific properties that characterize those classes of things.
  - E.g. as `rearSeatLegRoom` to describe a passenger vehicle
- In RDF Schema, properties are described using the RDF class `rdf:Property`, and the RDF Schema properties `rdfs:domain`, `rdfs:range`, and `rdfs:subPropertyOf`.
- All properties in RDF are described as instances of class `rdf:Property`.

```
exterms:weightInKg   rdf:type   rdf:Property .
```

Defining the `rdfs:range` 1/3

- RDF Schema also provides vocabulary for describing how properties and classes are intended to be used together in RDF data.
- The most important information of this kind is supplied by using the RDF Schema properties `rdfs:range` and `rdfs:domain` to further describe application-specific properties.
- The `rdfs:range` property is used to indicate that the values of a particular property are instances of a designated class.

```
ex:Person    rdf:type      rdfs:Class .
ex:author    rdf:type      rdf:Property .
ex:author    rdfs:range    ex:Person .
```

Defining the `rdfs:range` 2/3

- A property, say `ex:hasMother`, can have zero, one, or more than one range property.
- If `ex:hasMother` has no range property, then nothing is said about the values of the `ex:hasMother` property.
- If `ex:hasMother` has one range property, say one specifying `ex:Person` as the range, this says that the values of the `ex:hasMother` property are instances of class `ex:Person`.
- If `ex:hasMother` has more than one range property, say one specifying `ex:Person` as its range, and another specifying `ex:Female` as its range, this says that the values of the `ex:hasMother` property are resources that are instances of all of the classes specified as the ranges, i.e., that any value of `ex:hasMother` is both a `ex:Female` and a `ex:Person`.

Defining the `rdfs:range` 3/3

- The `rdfs:range` property can also be used to indicate that the value of a property is given by a typed literal.

```
ex:age    rdf:type       rdf:Property .
ex:age    rdfs:range     xsd:integer .
```

- The range declaration does not automatically "assign" a datatype to a plain literal in an instance RDF graph, and so a typed literal of the appropriate datatype must be explicitly provided.

`rdfs:Datatype`

- To explicitly state that `xsd:integer` is a datatype we can use `rdfs:Datatype`.

```
 xsd:integer   rdf:type   rdfs:Datatype .
```

- This statement says that `xsd:integer` is the URIref of a datatype.
- The statement does *not define* a datatype.
- This statement simply serves to document the existence of the datatype, and indicate explicitly that it is being used in this schema.

Defining the `rdfs:domain` 1/2

- The `rdfs:domain` property is used to indicate that a particular property applies to a designated class.
  - For example, if `example.org` wanted to indicate that the property `ex:author` applies to instances of class `ex:Book`, it would write the RDF statements:

```
ex:Book      rdf:type       rdfs:Class .
ex:author    rdf:type       rdf:Property .
ex:author    rdfs:domain    ex:Book .
```

- These statements indicate that `ex:Book` is a class, `ex:author` is a property, and that RDF statements using the `ex:author` property have instances of `ex:Book` as subjects.
- Example in RDF/XML:
  - http://www.w3.org/TR/rdf-primer/#example28

Defining the `rdfs:domain` 2/2

- A given property, say `exterms:weight`, may have zero, one, or more than one domain property.
- If `exterms:weight` has no domain property, then nothing is said about the resources that `exterms:weight` properties may be used with (any resource could have a `exterms:weight` property).
- If `exterms:weight` has one domain property, say one specifying `ex:Book` as the domain, this says that the `exterms:weight` property applies to instances of class `ex:Book`.
- If `exterms:weight` has more than one domain property, say one specifying `ex:Book` as the domain and another one specifying `ex:MotorVehicle` as the domain, this says that any resource that has a `exterms:weight` property is an instance of all of the classes specified as the domains, i.e., that any resource that has a `exterms:weight` property is both a `ex:Book` and a `ex:MotorVehicle`.

Defining `rdf:subPropertyOf`

- RDF Schema provides a way to specialize properties as well as classes.
- For example, if `ex:primaryDriver` and `ex:driver` are both properties, we could describe these properties, and the fact that `ex:primaryDriver` is a specialization of `ex:driver`, by writing the RDF statements:

```
ex:driver            rdf:type              rdf:Property .
ex:primaryDriver     rdf:type              rdf:Property .
ex:primaryDriver     rdfs:subPropertyOf    ex:driver .
```

- The meaning of this `rdfs:subPropertyOf` relationship is that if an instance `exstaff:fred` is an `ex:primaryDriver` of the instance `ex:companyVan`, then RDF Schema defines `exstaff:fred` as also being an `ex:driver` of `ex:companyVan`.

RDF Schema build in Properties

- `rdfs:comment`
  - May be used to provide a human-readabel description of a resource.
- `rdfs:label`
  - May be used to provide a human-readabel version of a resource's name.
- `rdfs:seeAlso`
  - Is used to indicate a resource that might provide additional information about the subject resource.
- `rdfs:isDefinedBy`
  - Is used to indicate a resource defining the subject resource. This property may be used to indicate an RDF vocabulary in which a resource is described.

Naming Convention

- Class names are written with an initial uppercase letter.
- Property and instance names are written with an initial lowercase letter.
- This convention is not required in RDF Schema, but is good style.

Interpreting RDF Schema Declarations: Comparison to OO-Programming

- The RDF Schema type system is similar to the type systems of object-oriented programming languages such as Java.
- However, RDF differs from most programming language type systems in several important respects.
- One important difference is that instead of describing a class as having a collection of specific properties, an RDF Schema describes properties as applying to specific classes of resources, using domain and range properties.
  - For example, a typical object-oriented programming language defines a class `Book` with an attribute called `author` having values of type `Person`.
  - A corresponding RDF Schema describes a class `ex:Book`, and, in a separate description, a property `ex:author` having a domain of `ex:Book` and a range of `ex:Person`.
  - In RDF Schema everyone can add properties to a class.

Scope of Properties 1/2

- The scope of an attribute description in most programming languages is restricted to the class or type in which it is defined.
  - In the programming language class description, the attribute `author` is part of the description of class `Book`, and applies only to instances of class `Book`. Another class (say, `SoftwareModule`) might also have an attribute called `author`, but this would be considered a different attribute.
  - In RDF property descriptions are, by default, independent of class definitions, and have, by default, global scope (although they may optionally be declared to apply only to certain classes using domain specifications).
- As a result, an RDF schema could describe a property `exterms:weight` without a domain being specified.
- This property could then be used to describe instances of any class that might be considered to have a weight. One benefit of the RDF property-based approach is that it becomes easier to extend the use of property definitions to situations that might not have been anticipated in the original description. At the same time, this is a "benefit" which must be used with care, to insure that properties are not mis-applied in inappropriate situations.

Scope of Properties 2/2

- Another result of the global scope of RDF property descriptions is that it is not possible in an RDF Schema to define a specific property as having locally-different ranges depending on the class of the resource it is applied to.
  - For example, in defining the property `ex:hasParent`, it would be desirable to be able to say that if the property is used to describe a resource of class `ex:Human`, then the range of the property is also a resource of class `ex:Human`, while if the property is used to describe a resource of class `ex:Tiger`, then the range of the property is also a resource of class `ex:Tiger`. This kind of definition is not possible in RDF Schema. Instead, any range defined for an RDF property applies to all uses of the property, and so ranges should be defined with care.

Interpreting RDF Schema Declarations 1/4

- RDF Schema descriptions are not necessarily prescriptive in the way programming language type declarations typically are.
  - For example, if a programming language declares a class `Book` with an `author` attribute having values of type `Person`, this is usually interpreted as a group of constraints.
  - The language will not allow the creation of an instance of `Book` without an author attribute, and it will not allow an instance of `Book` with an `author` attribute that does not have a `Person` as its value.
  - Moreover, if `author` is the only attribute defined for class `Book`, the language will not allow an instance of `Book` with some other attribute.
- RDF Schema, provides information as additional descriptions of resources, but does not prescribe how these descriptions should be used by applications.

Interpreting RDF Schema Declarations 2/4

- Different interpretations are possible for the following schema.

```
ex:author  rdf:type     rfd:Property .
ex:author  rdfs:range  ex:Person .
```

- One application might interpret these statements as specifying part of a template for RDF data it is creating, and use it to ensure that any `ex:author` property has a value of the indicated (`ex:Person`) class. That is, this application interprets the schema description as a constraint in the same way that a programming language might.

Interpreting RDF Schema Declarations 3/4

- Another application might interpret this statement as providing additional information about data it is receiving, information which may not be provided explicitly in the original data. For example, this application might receive some RDF data that includes an `ex:author` property whose value is a resource of unspecified class, and use this schema-provided statement to conclude that the resource must be an instance of class `ex:Person`.
- A third application might receive some RDF data that includes an `ex:author` property whose value is a resource of class `ex:Corporation`, and use this schema information as the basis of a warning that "there may be an inconsistency here, but on the other hand there may not be". Somewhere else there may be a declaration that resolves the apparent inconsistency (e.g., a declaration to the effect that "a Corporation is a (legal) Person").

Interpreting RDF Schema Declarations 4/4

- Moreover, depending on how the application interprets the property descriptions, a description of an instance might be considered valid either without some of the schema-specified properties (e.g., there might be an instance of `ex:Book` without an `ex:author` property, even if `ex:author` is described as having a domain of `ex:Book`), or with additional properties (there might be an instance of `ex:Book` with an `ex:technicalEditor` property, even though the schema describing class `ex:Book` does not describe such a property).
- In other words, statements in an RDF schema are always descriptions. They may also be prescriptive (introduce constraints), but only if the application interpreting those statements wants to treat them that way. All RDF Schema does is provide a way of stating this additional information. Whether this information conflicts with explicitly specified instance data is up to the application to determine and act upon.

Writing RDF Schema Ontologies

- Ontology Editors:
  - Protègè + OWL plug-in (current version supports OWL only): http://protege.stanford.edu/
  - SWOOP: http://www.mindswap.org/2004/SWOOP/
  - TopBraid Composer: http://www.topquadrant.com/products/TB_Composer.html