Overview

- RDF is used to specify semantics for data expressed in XML in a standardized, interoperable manner
- RDF is a simple frame language
  - Data is described in relational sentences using binary relations
  - Classes and relations are defined in schema
  - Classes are organized in subclass hierarchies
  - Domain and range constraints on relations are supported
- Provides interoperability between applications that exchange machine-understandable information on the Web
RDF Statements

- **Resource**
  - Ñ Object in the domain of discourse
    - E.g., Web page, portion of Web page, Web site
  - Ñ Denoted by *resource identifier* (i.e., object constant)

- **Property**
  - Ñ Binary relation in the domain of discourse
    - Used to describe a resource
  - Ñ Relates resources to other resources or to *literals* (constants)

- **Statement**
  - Ñ Relational sentence: *(Property Subject Object)*
    - I.e., a slot value
  - Ñ Subject: a resource
  - Ñ Object: resource identifier or XML primitive data type

---

RDF Example

*Ora Lassila is the creator of the resource http://www.w3.org/Lassila.*

- **RDF Statement:**
  - Subject:  http://www.w3.org/Lassila
  - Property:  Creator
  - Object:  “Ora Lassila”

- **KIF equivalent:**
  - Ñ (Creator http://www.w3.org/Lassila “Ora Lassila”)
RDF Description

**Description**
- A group of statements about the same resource (i.e., a frame)
- RDF element
  - Wrapper that marks the boundaries in an XML document of content intended to be mappable into a set of RDF statements
  - Syntax: RDF ::= `<RDF attributeSpec* '>`
    descriptionElement* `'</RDF>'`
- Description element
  - Identifies the resource being described
  - Syntax:
    description ::= `<rdf:Description` idAboutAttr? `'>`
    propertyElement* `'</rdf:Description>'`
    idAboutAttr ::= idAttr | aboutAttr
    aboutAttr ::= 'about="" URI-reference '"'
    idAttr ::= 'ID= "" IDsymbol '"'

**Property element**
- A property-value pair corresponding to an RDF statement
- Syntax
  propertyElement ::= `<' propName '>' value '</' propName '>'`
  value ::= description | string
  resourceAttr ::= 'resource="" URI-reference '"'

**Example description**

Ora Lassila is the creator of the resource http://www.w3.org/Lassila

```xml
<RDF xmlns = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:s = "http://description.org/schema/">
  <Description about = "http://www.w3.org/Lassila">
    <s:Creator> Ora Lassila </s:Creator>
  </Description>
</RDF>
```
RDF Example

“Ora Lassila is the creator of the resource http://www.w3.org/Lassila.”

◆ RDF Description:

```xml
<RDF xmlns = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:s = "http://description.org/schema/">
  <Description about="http://www.w3.org/Lassila">
    <s:Creator> Ora Lassila </s:Creator>
  </Description>
</RDF>
```

◆ OKBC equivalent:

http://www.w3.org/Lassila
Creator: “Ora Lassila”

◆ RDF Statement:

Subject: http://www.w3.org/Lassila
Property: Creator
Object: “Ora Lassila”

Example RDF Description

```xml
<RDF xmlns:dc = "http://purl.org/metadata/dublin_core#">
  <Description about = "http://www.dlib.org">
    <dc:Title> D-Lib Program </dc:Title>
    <dc:Description> The D-Lib program ... </dc:Description>
    <dc:Publisher> Corporation For National ... </dc:Publisher>
    <dc:Date> 1995-01-07 </dc:Date>
    <dc:Subject>
      <Bag> <li> Research; statistical methods </rdf:li>
           <li> Education, research, related topics </rdf:li>
           <li> Library use Studies </rdf:li>
      </Bag>
    </dc:Subject>
    <dc:Type> World Wide Web Home Page </dc:Type>
    <dc:Format> text/html </dc:Format>
    <dc:Language> en </dc:Language>
  </Description>
</RDF>
```
**OKBC Equivalent of Example**

http://www.dlib.org
Title: “D-Lib Program”
Description: “The D-Lib program …”
Publisher: “Corporation For National …”
Date: “1995-01-07”
Subject: (listof “Research; statistical methods”
“Education, research, related topics”
“Library use Studies”)
Type: “World Wide Web Home Page”
Format: “text/html”
Language: “en”

**RDF Ontology**

- Classes
  - Resource
  - Property
  - Class
  - Literal
  - Statement
  - Container
    - Bag
    - Seq
    - Alt
    - ContainerMembershipProperty

- Properties
  - type
  - subject
  - predicate
  - object
  - value
  - _1, _2, _3, …
RDF Ontology (as OKBC Frames)

Resource
- Documentation: “The class of all objects.”
- Instance-Of: Class
- Disjoint-Decomposition: (listof Class Property)

Class
- Documentation: “The class of all classes.”
- Subclass-Of: Resource
- Instance-Of: Class

Property
- Documentation: “The class of all properties.”
- Subclass-Of: Resource
- Instance-Of: Class

type
- Documentation: “Class membership property.”
- Instance-Of: Property

RDF Schema

- An ontology added to RDF
  - Classes [2]
    - ConstraintResource
    - ConstraintProperty
  - Properties [8]
    - subClassOf
    - subPropertyOf
    - seeAlso
    - isDefinedBy
    - comment
    - label
    - range
    - domain
RDF-S Ontology (as OKBC frames)

- subClassOf
  Documentation: “The subset/superset relation between classes.”
  Instance-Of: Property

- subPropertyOf
  Documentation: “Specifies that one property is a specialization of another.”
  Instance-Of: Property

- seeAlso
  Documentation: “Specifies a resource that contains information about the subject resource.”
  Instance-Of: Property

- isDefinedBy
  Documentation: “Specifies the resource defining the subject resource.”
  Instance-Of: Property
  subPropertyOf: seeAlso

RDF Classes (using RDF-S properties)

- Resource
  Documentation: “The class of all objects.”
  type: Class
  Disjoint-Decomposition: (listof Class Property)

- Class
  Documentation: “The class of all classes.”
  subClassOf: Resource
  type: Class

- Property
  Documentation: “The class of all properties.”
  subClassOf: Resource
  type: Class
**RDF-S Class and Property Definitions**

- `<rdf:Description ID = "MotorVehicle">
  <rdf:type resource = "http://www.w3.org/PR-rdf-schema-19990303#Class" />
  <rdfs:subClassOf rdf:resource = "http://www.w3.org/PR-rdf-schema-19990303#Resource" />
</rdf:Description>

- `<rdf:Description ID = "PassengerVehicle">
  <rdf:type resource = "http://www.w3.org/PR-rdf-schema-19990303#Class" />
  <rdfs:subClassOf rdf:resource = "#MotorVehicle" />
</rdf:Description>

- `<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>

**Compact Syntax For Instances**

- A class name can be used as an element name
- Example property definition:
  `<rdf:Description ID = "biologicalParent">
   <rdf:type resource = "www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  </rdf:Description>
  can be written as:
  `<rdf:Property ID = "biologicalParent" />`
- Example class definition:
  `<rdf:Description ID = "MotorVehicle">
   <rdf:type resource = "http://www.w3.org/PR-rdf-schema-19990303#Class" />
   <rdfs:subClassOf rdf:resource = "http://www.w3.org/PR-rdf-schema-19990303#Resource" />
  </rdf:Description>
  can be written as:
  `<rdf:Class ID = "MotorVehicle">
   <rdfs:subClassOf rdf:resource = "http://PR-rdf-schema-19990303#Resource" />
  </rdf:Class>`
Example Compact Definitions

```xml
<rdf:Class ID="MotorVehicle">
</rdf:Class>

<rdf:Class ID="PassengerVehicle">
  <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
</rdf:Class>

<rdf:Class ID="Van">
  <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
</rdf:Class>

<rdf:Class ID="MiniVan">
  <rdfs:subClassOf rdf:resource="#Van"/>
  <rdfs:subClassOf rdf:resource="#PassengerVehicle"/>
</rdf:Class>
```

RDF-S Constraints

- Domain and range specification on properties
- Examples:

```xml
<rdf:Description ID = "registeredTo">
  <rdf:type resource = "www.w3.org/ ... rdf-syntax-ns#Property" />
  <rdfs:domain rdf:resource = "#MotorVehicle" />
  <rdfs:range rdf:resource = "#Person" />
</rdf:Description>

<rdf:Description ID = "rearSeatLegRoom">
  <rdf:type resource = "www.w3.org/ ... rdf-syntax-ns#Property" />
  <rdfs:domain rdf:resource = "#PassengerVehicle" />
  <rdfs:domain rdf:resource = "#Minivan" />
  <rdfs:range rdf:resource = "www.w3.org/datatypes#Number" />
</rdf:Description>
```
Example Class Definition

```xml
<rdf:RDF xmlns:rdf = "www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs = "//www.w3.org/TR/1999/PR-rdf-schema-19990303#">

  <rdfs:Class rdf:ID = "Person">
    <rdfs:comment> The class of people. </rdfs:comment>
    <rdfs:subClassOf rdf:resource =
                     "www.classstypes.org/useful_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.datatypes.org/useful_types#Integer" />
    <rdfs:domain rdf:resource = "#Person" />
  </rdf:Property>

  <rdf:Property ID = "age">
    <rdfs:range rdf:resource =
                "http://www.datatypes.org/useful_types#Integer" />
    <rdfs:domain rdf:resource = "#Person" />
  </rdf:Property>

</rdf:RDF>
```

Example Continued

```xml
<rdf:Property ID = "ssn">
  <rdfs:comment> Social Security Number </rdfs:comment>
  <rdfs:range rdf:resource =
              "http://www.datatypes.org/useful_types#Integer" />
  <rdfs:domain rdf:resource = "#Person" />
</rdf:Property>

<rdf:Property ID = "age">
  <rdfs:range rdf:resource =
              "http://www.datatypes.org/useful_types#Integer" />
  <rdfs:domain rdf:resource = "#Person" />
</rdf:Property>
```
Example Concluded

```xml
<rdfs:Class rdf:ID = "MaritalStatus" />

<MaritalStatus rdf:ID = "Married" />  
<MaritalStatus rdf:ID = "Divorced" />  
<MaritalStatus rdf:ID = "Single" />  
<MaritalStatus rdf:ID = "Widowed" />  

</rdf:RDF>
```

Comments on RDF and RDF-S

- Severely lacking in expressive power
  - Domain and range constraints rather than Value-Type
    - E.g., can’t define class of people all of whose children are male
  - No cardinality constraints
    - Particularly important for “exactly 1” and “at most 1”
  - No decompositions
    - Particularly important for “disjoint” and “exhaustive”
  - No axioms
  - No negation (!)
- Not useful for checking consistency
  - E.g., can’t prove an object is not an instance of a class
- Basically a typing system
- Does provide identifiers for objects
  - Enables referencing and describing