RDF (Resource Description Framework)

1. RDF provides a way of describing resources via metadata (data about data)
   It restricts the description of resources to triplets (subject,predicate,object)
1. It provides interoperability between applications that exchange machine understandable information on the Web.
3. The broad goal of RDF is to define a mechanism for describing resources that makes no assumptions about a particular application domain, nor defines (a priori) the semantics of any application domain.

- Uses XML as the interchange syntax.
- Provides a **lightweight** ontology system.

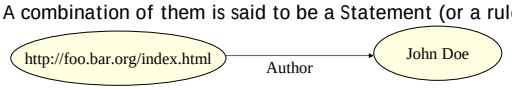
The formal specification of RDF is available at:
http://www.w3.org/TR/REC-rdf-syntax/

RDF Syntax

Subject, Predicate and Object Triplets (Tuples)

- Subject: The resource being described.
- Predicate: A property of the resource
- Object: The value of the property

A combination of them is said to be a Statement (or a rule)



A web page being described	A property of the web page (author)	The value of the predicate (here the author)
[Subject]	[Predicate]	[Object]

RDF Example

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
    xmlns:s="http://description.org/schema/">

  <rdf:Description about="http://foo.bar.org/index.html">
     <s:Author>John Doe</s:Author>
  </rdf:Description>
</rdf:RDF>
```

Namespace for the RDF spec

Namespace 's', a custom namespace

Subject

Author (property of the subject) (Also a resource)

Object. Can also point to a resource

The above statement says :
The Author of http://foo.bar.org/index.html is "John Doe"

In this way, we can have different objects (resources) pointing to other objects (resources) , thus forming a DLG (Directed Line Graph)

You can also make statements about statements – reification
Ex: 'xyz' says that ' The Author of http://foo.bar.org/index.html is John Doe'

RDF Schema

- A schema defines the terms that will be used in the RDF statements and gives specific meanings to them.
  `http://www.w3.org/TR/rdf-schema/`

Example:

```
<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="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 Schema Namespace

An "ID" attribute actually defines a new resource

"Resource" is the top level class

PassengerVehicle is a subclass of MotorVehicle

Example (cont..)

```
<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: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>
```

Multiple Inheritance

Domain of a property

Range of a property

RDF: Tools/Resources

SirPAC
A Simple RDF Parser & Compiler. It parses the RDF, and validates it. It also generates the tuples and even draws a graph of the data model.
www.w3.org/RDF/Implementations/SiRPAC/

Reggie
A Nice Metadata Editor. Java based simple user interface to describe a web resource. Can mail the metadata file to yourself after finished editing.
http://metadata.net/dstc/

Protégé
Editor of ontologies in practically any language you care about. Open source.
http://www.smi.stanford.edu/projects/protege/

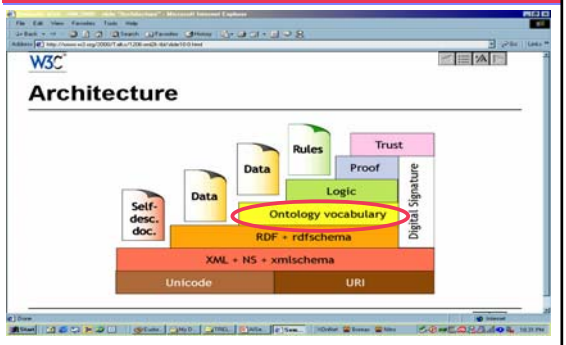
Summary: RDF & RDF Schema layer

- Minimalist model - (thing), Class, Property
- Subproperty, Subclass
- Domain & Range

- Still not a W3C recommendation
- Continues to change

- Other languages are being built on XML substrate: XQUERY, XTM

The Layer Cake [TBL,XML2000]



Limitations of RDF

- Cannot define properties of properties (unique, transitive)

- No equivalence, disjointness, etc.

- No mechanism of specifying **necessary and sufficient conditions** for class membership.
  Example:
  If it is given that 'XYZ' has a 'car' which is '7ft high', has 'wide wheels' and 'loading space is 4 cub.m', then we should be able to reason that 'XYZ' has an 'SUV', as given by the necessary and sufficient conditions for being an 'SUV' :
  height > 4ft & wide wheels & loading space > 2 cub.m

DAML+OIL's History

- **W3C**'s Semantic Web Activity:
  - RDF and metadata markup efforts to represent data in a machine understandable form.
- **DARPA** started the DARPA Agent Markup Language (**DAML**) program.
  - possibly with "ARPANET -> Internet" in mind
- **EC** (European Commission) funding programs
  - Ontology Interchange Language (**OIL**)
    - logic based language.
    - brings logic and inference to the Semantic Web

www.daml.org
DAML+OIL: http://www.daml.org/2001/03/daml+oil-index.html

DAML+OIL (www.daml.org)

- It builds on earlier W3C standards such as RDF and RDF Schema.
- DAML extends RDF and RDFS with richer modelling primitives.
  - disjointWith, intersectionOf, oneOf, cardinality
- Able to provide properties of properties
  - uniqueness, transitivity, etc.
- Current version DAML+OIL provides a semantic interpretation (model-theoretic semantics)

http://www.daml.org/2001/03/daml+oil-index.html

An Example (from www.daml.org)

```
<rdf:RDF
 xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:daml="http://www.daml.org/2000/12/daml+oil#"
 xmlns     ="http://www.daml.org/2000/12/daml+oil-ex#"
>

<daml:Ontology about=""> ——→ Start of an ontology (about = "" implies 'this' document)
<daml:versionInfo>An example ontology</daml:versionInfo>

<rdfs:Class rdf:ID="Animal">
 <rdfs:label>Animal</rdfs:label> ——→ The label is not used for logical interpretation
 <rdfs:comment>
  This class of animals is illustrative of a number of ontological idioms.
 </rdfs:comment>
</rdfs:Class>

<rdfs:Class rdf:ID="Male">
 <rdfs:subClassOf rdf:resource="#Animal"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Female">
 <rdfs:subClassOf rdf:resource="#Animal"/>
 <daml:disjointWith rdf:resource="#Male"/> ——→ Can explicitly specify the set of Females to
</rdfs:Class>                                  be disjoint with the set of Males

<rdfs:Class rdf:ID="Man">                  ——→ The Person class is defined later
 <rdfs:subClassOf rdf:resource="#Person"/>
 <rdfs:subClassOf rdf:resource="#Male"/>   ——→ To be read conjunctively. A man is a sub-class of
</rdfs:Class>                                  'Person' and a 'Male'
```

Example (contd..)

```
<rdfs:Class rdf:ID="Woman">
  <rdfs:subClassOf rdf:resource="#Person"/>
  <rdfs:subClassOf rdf:resource="#Female"/>
</rdfs:Class>

<rdf:ObjectProperty rdf:ID="hasParent">
  <rdfs:domain rdf:resource="#Animal"/>
  <rdfs:range rdf:resource="#Animal"/>
</rdf:ObjectProperty>

<rdf:ObjectProperty rdf:ID="hasFather">
  <rdfs:subPropertyOf rdf:resource="#hasParent"/>
  <rdfs:range rdf:resource="#Male"/>
</rdf:ObjectProperty>

<daml:DatatypeProperty rdf:ID="age">
   <rdfs:range rdf:resource="http://www.w3.org/2000/10/XMLSchema#nonNegativeInteger"/>
</daml:DatatypeProperty>

<rdfs:Class rdf:ID="Person">
  <rdfs:subClassOf rdf:resource="#Animal"/>
  <rdfs:subClassOf>
    <daml:Restriction>
      <daml:onProperty rdf:resource="#hasParent"/>
      <daml:toClass rdf:resource="#Person"/>
    </daml:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <daml:Restriction daml:cardinality="1">
      <daml:onProperty rdf:resource="#hasFather"/>
    </daml:Restriction>
  </rdfs:subClassOf>
</rdfs:Class>
```

An objectProperty relates objects to objects

Describes the element which encloses this Property

Describes the value of the Property

Note: Contrary to RDF, DAML takes the 'intersection' of the domains/ranges if multiple domains/ranges are specified

A datatype property relates an object to a primitive datatype value

The XML Schema datatype is referenced here

The Restriction defines an anonymous class of all things that satisfy the restriction.

Restrictions on the property hasParent (only for the Person class – Local scope, as opposed to rdfs:range)
A person can have only another Person as it's parent

A Person can have only 1 Father

Example (contd..)

```
<rdfs:Class rdf:about="#Animal">
<rdfs:subClassOf>
  <daml:Restriction daml:cardinality="2">
    <daml:onProperty rdf:resource="#hasParent"/>
  </daml:Restriction>
</rdfs:subClassOf>
</rdfs:Class>

<rdfs:Class rdf:about="#Person">
  <rdfs:subClassOf>
    <daml:Restriction daml:maxcardinality="1">
      <daml:onProperty rdf:resource="#hasSpouse"/>
    </daml:Restriction>
  </rdfs:subClassOf>
</rdfs:Class>
```

Addition to the Animal Class without modifying it -- "about"

Restrictions on the property hasParent
An animal can have exactly 2 parents

Restrictions on the property hasSpouse
A person can have only 1 spouse

Further constructs that the example doesn't use :

Properties:
- TransitiveProperty (hasAncestor), UniqueProperty (hasMother), inverseOf(hasChild -> hasParent), etc.

Classes:
- intersectionOf (a daml:collection), unionOf (a daml:collection), sameClassAs, complementOf, etc.

DAML References/Tools

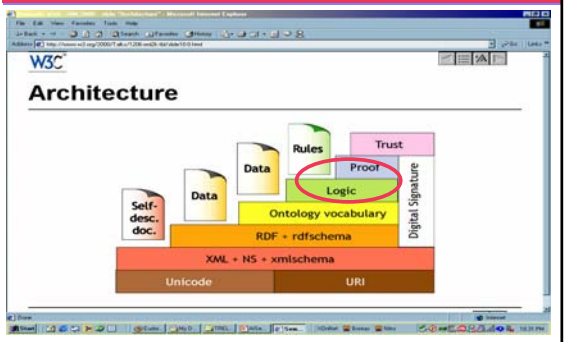
DAML Viewer:
It provides a means to view the instances found in a DAML document.
http://www.daml.org/viewer/applet.html

DAML Crawler Results:
A list of .daml files on the internet
http://www.daml.org/crawler/pages.html

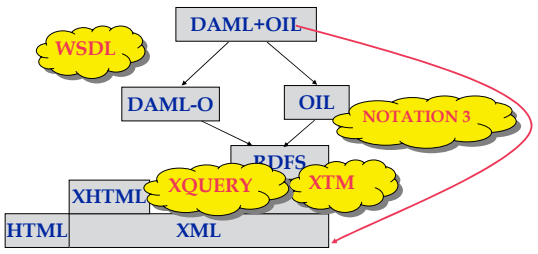
A DAML Validator
http://www.daml.org/validator/

A DAML example explained:
It has the same example as in the slides, discussed in detail.
http://www.daml.org/2001/03/daml+oil-walkthru.html

The Layer Cake [TBL,XML2000]



The "Layer" Cake



W3C's Semantic Web Principles

1. Everything identifiable is in the Semantic Web (URIs!)
2. Partial information
   - Anyone can say anything about anything
3. Web of trust
   - All statements on the Web occur in some context
4. Evolution
   - Allow combining independent work done by different communities
5. Minimalist design
   - Make the simple things simple, and the complex things possible
   - Standardize no more than is necessary

Hypertext: Then and Now

- SOTA circa 1990: Dynatext's electronic book
  - A book had to be compiled (like a program) in order to be displayed efficiently
  - A central link database, to make sure there were no broken links
  - Text that was fixed and consistent (a whole book)
- WWW:
  - Links can be added and used at any time
  - Distributed (must live with broken links!)
  - Decentralized

Knowledge Representation: Now and Tomorrow

"To webize KR in general is, in many ways, the same as to webize hypertext. Replace identifiers with URIs. Remove any requirement for global consistency. Put any significant effort into getting critical mass. Sit back."

-- TBL

Ongoing Work at ISI

- EXPECT (k acquisition and problem solving)
  - No longer developing KBs, but importing schemas and data
- Electric Elves
  - Agents are more transparent and publish data & schemas, advertisements/assumptions
- TRELLIS (try it out at trellis.semanticweb.org!)
  - Users represent decisions and opinions -> Web of Trust
- IKRAFT
  - Users turn text in progressively more formal representations (KB) -> semi-formal annotations