An overview of

Semantic Web Languages and Technologies

Semantic Web Technologies

- W3C "recommendations"
  - RDF, RDFS, RDFa, OWL, SPARQL, …
  - Under development: recommendations for rules, provenance, database export, etc.
- Common tools and systems -- commercial, free and open sourced
  - Ontology editors, triple stores, reasoners, etc.
- Common ontologies and data sets
  - Foaf, Dbpedia, SKOS, etc.
- Infrastructure systems
  - Search, ontology metadata, linking services

Common KR languages

- Knowledge representation (KR) has always been an important part of AI and other disciplines
- Many approaches have been developed, implemented and evolved since the 1960s
- Most were one-offs, used only by the developers
- Starting in the 1990s, there was an interest in developing a common KR language to support knowledge reuse and distributed KB systems
- The Semantic Web languages are the current generation of this idea

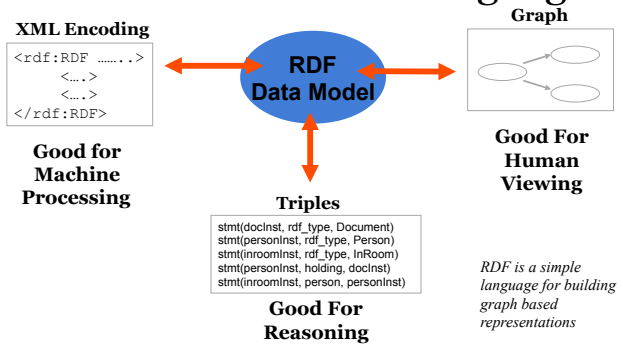
Semantic web languages today

- Today there are, IOHO, two semantic web languages
  - **RDF** – Resource Description Framework http://www.w3.org/RDF/
  - **DAML+OIL** – Darpa Agent Markup Language http://www.daml.org/ (deprecated)
  - **OWL** – Ontology Web Language http://www.w3.org/2001/sw/
- **Topic maps** (http://topicmaps.org/) are another species, not based on RDF
- Microformats, Common Logic, etc. offer other possibilities

Two Semantic Web Notions

- **The semantic web**
  - Idea of a web of machine understandable information
  - Agnostic about the technology used to support it
  - May involve more AI (e.g., NLP)
  - Human end users in the center
- **The Semantic Web**
  - The current vision of a semantic web as defined by the W3C community: a web of data
  - Using W3C supported standards, i.e., RDF, OWL, SPARQL, XML, RIF, etc.
  - By machines for machines with human oriented applications on top

RDF is the first SW language



The RDF Data Model

- An RDF document is an unordered collection of statements, each with a **subject**, **predicate** and **object** (aka **triples**)
- A triple can be thought of as a labelled arc in a graph
- Statements describe properties of web **resources**
- A resource is any object that can be pointed to by a **URI**:
  - a document, a picture, a paragraph on the Web, …
  - E.g., http://umbc.edu/~finin/cv.html
  - a book in the library, a real person (?)
  - isbn://5031-4444-3333
- Properties themselves are also resources (URIs)

URIs are a foundation

- URI = **Uniform Resource Identifier**
  - "The generic set of all names/addresses that are short strings that refer to resources"
  - URLs (Uniform Resource Locators) are a subset of URIs, used for resources that can be *accessed* on the web
- URIs look like "normal" URLs, often with fragment identifiers to point to a document part:
  - http://foo.com/bar/mumble.html#pitch

URIs are a foundation

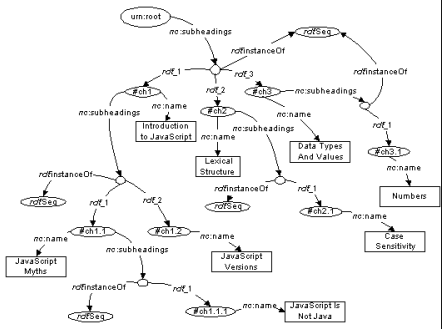
- URIs are unambiguous, unlike natural language terms -- the web provides a global **namespace**
- A URI denotes something, e.g., a concept, entity or relation
- We usually assume references to the same URI are to the same thing

What does a URI mean?

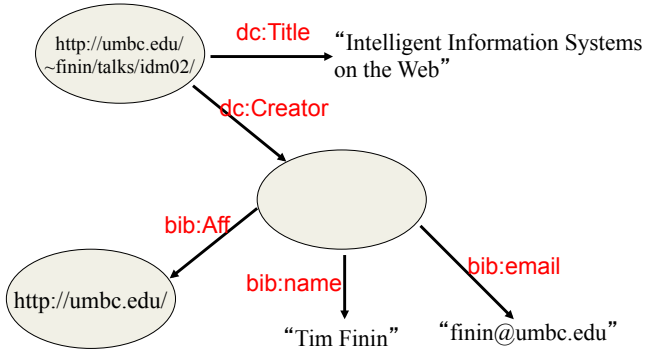
- Sometimes URIs denote a web resource
  - http://umbc.edu/~finin/finin.jpg denotes a file
  - We can use RDF to make assertions about the resource, e.g., it's an image and depicts a person with name Tim Finin, …
- Sometimes concepts in the external world
  - E.g., http://umbc.edu/ denotes a particular University located in Baltimore
  - This is done by social convention
- Cool URIs don't change
  - http://www.w3.org/Provider/Style/URI

The RDF Graph

- An RDF document is an unordered collection of triples
- The subject of one triple can be the object of another
- So the result is a directed, labelled graph
- A triple's object can also be a literal, e.g., a string.



Simple RDF Example



Serialization

- A graph is an abstract model, we'll need to serialize it as text for many reasons, e.g., display, editing, exchange
- There are several standard RDF serializations, the three most important are: XML, turtle and ntriples
- Most Semantic Web tools can read or write in any of these serializations

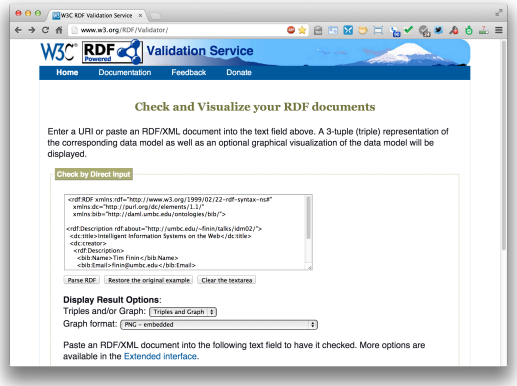
XML encoding for RDF

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:dc="http://purl.org/dc/elements/1.1/"
   xmlns:bib=http://daml.umbc.edu/ontologies/bib/>
<rdf:Description rdf:about="http://umbc.edu/~finin/talks/idm02/">
  <dc:title>Intelligent Information Systems on the Web</dc:title>
  <dc:creator>
    <rdf:Description>
      <bib:Name>Tim Finin</bib:Name>
      <bib:Email>finin@umbc.edu</bib:Email>
      <bib:Aff rdf:resource="http://umbc.edu/" />
    </rdf:Description>
  </dc:creator>
</rdf:Description>
</rdf:RDF>
```

Note the prefix declarations

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:dc="http://purl.org/dc/elements/1.1/"
   xmlns:bib=http://daml.umbc.edu/ontologies/bib/>
<rdf:Description rdf:about="http://umbc.edu/~finin/talks/idm02/">
  <dc:title>Intelligent Information Systems on the Web</dc:title>
  <dc:creator>
    <rdf:Description>
      <bib:Name>Tim Finin</bib:Name>
      <bib:Email>finin@umbc.edu</bib:Email>
      <bib:Aff rdf:resource="http://umbc.edu/" />
    </rdf:Description>
  </dc:creator>
</rdf:Description>
</rdf:RDF>
```

An RDF validation service



http://www.w3.org/RDF/Validator/

Easy to convert between serializations

- Most software tools can read and write different serilizations
- rdf2rdf is a simple utility that's handy

N-triple representation

- RDF can be encoded as a set of **triples**.
  <subject> <predicate> <object> .

<http://umbc.edu/~finin/talks/idm02/> <http://purl.org/dc/elements/1.1/title> "Intelligent Information Systems on the Web" .
<http://umbc.edu/~finin/talks/idm02/> <http://purl.org/dc/elements/1.1/creator> _:node17i6ht38ux1 .
_:node17i6ht38ux1 <http://daml.umbc.edu/ontologies/bib/Name> "Tim Finin" .
_:node17i6ht38ux1 <http://daml.umbc.edu/ontologies/bib/Email> "finin@umbc.edu" .
_:node17i6ht38ux1 <http://daml.umbc.edu/ontologies/bib/Aff> <http://umbc.edu/> .

N3/Turtle notation for RDF

- N3 is a compact notation for RDF that is easier for people to read, write and edit
- It's just "syntactic sugar"
- Aka Notation 3, developed by TBL himself
- But, XML is largely unreadable and even harder to write
- Turtle is a W3C standard that covers most of N3

N3 Example

```
@prefix rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns# .
@prefix dc: http://purl.org/dc/elements/1.1/ .
@prefix bib: http://daml.umbc.edu/ontologies/bib/ .

<http://umbc.edu/~finin/talks/idm02/>
  dc:title "Intelligent Information Systems on the Web" ;
 dc:creator
    [ bib:Name "Tim Finin";
      bib:Email "finin@umbc.edu"
      bib:Aff: "http://umbc.edu/" ] .
```

Triple Notes

- **RDF triples have one of two forms:**
  - <URI> <URI> <URI>
  - <URI> <URI> <quoted string>
- **Triples are also easily mapped into logic**
  - <subject> <predicate> <object> becoming:
    - <predicate>(<subject>,<object>)
    - With type(<S>,<O>) becoming <O>(<S>)
  - Example:
    - subclass(man,person)
    - sex(man,male)
    - domain(sex,animal)
    - man(adam)
    - age(adam,100)

; Note: we're not
; showing the actual
; URIs for clarity

- **Triples are easily stored and managed in DBMS**
  - Flat nature of a triple a good match for relational DBs

A usecase: FOAF

- FOAF (Friend of a Friend) is a simple ontology to describe people and their social networks.
  - See the foaf project page: http://www.foaf-project.org/
- In 2008 we crawled the web and discovered over 1,000,000 valid RDF FOAF files.
  - Most of these are from the http://liveJournal.com/ blogging system which encodes basic user info in foaf
  - See http://apple.cs.umbc.edu/semdis/wob/foaf/

```
<foaf:Person>
  <foaf:name>Tim Finin</foaf:name>
  <foaf:mbox_sha1sum>2410…37262c252e</foaf:mbox_sha1sum>
  <foaf:homepage rdf:resource="http://umbc.edu/~finin/" />
  <foaf:img rdf:resource="http://umbc.edu/~finin/images/passport.gif" />
</foaf:Person>
```

FOAF Vocabulary

Basics
Agent
Person
name
nick
title
homepage
mbox
mbox_sha1sum
img
depiction (depicts)
surname
family_name
givenname
firstName

Personal Info
weblog
knows
interest
currentProject
pastProject
plan
based_near
workplaceHomepage
workInfoHomepage
schoolHomepage
topic_interest
publications
geekcode
myersBriggs
dnaChecksum

Documents & Images
Document
Image
PersonalProfileDocument
topic (page)
primaryTopic
tipjar
sha1
made (maker)
thumbnail
logo

Online Accts
OnlineAccount
OnlineChatAccount
OnlineEcommerceAccount
OnlineGamingAccount
holdsAccount
accountServiceHomepage
accountName
icqChatID
msnChatID
aimChatID
jabberID
yahooChatID

Projects & Groups
Project
Organization
Group
member
membershipClass
fundedBy
theme

FOAF: why RDF? Extensibility!

- FOAF vocabulary provides 50+ basic terms for making simple claims about people
- FOAF files can use other RDF terms too: RSS, MusicBrainz, Dublin Core, Wordnet, Creative Commons, blood types, starsigns, …
- RDF guarantees freedom of independent extension
  - OWL provides fancier data-merging facilities
- **Result:** Freedom to say what you like, using any RDF markup you want, and have RDF crawlers merge your FOAF documents with other's and know when you're talking about the same entities.

After Dan Brickley, danbri@w3.org

No free lunch!

Consequence:

- We must plan for lies, mischief, mistakes, stale data, slander
- Dataset is out of control, distributed, dynamic
- Importance of knowing who-said-what
  - Anyone can describe anyone
  - We must record data provenance
  - Modeling and reasoning about trust is critical
- Legal, privacy and etiquette issues emerge
- Welcome to the real world

After Dan Brickley, danbri@w3.org

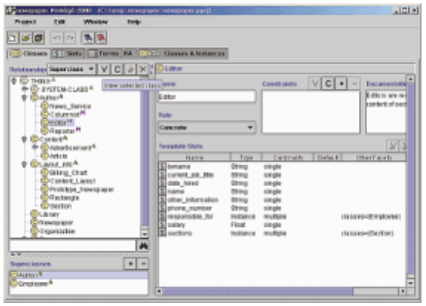
More RDF Vocabulary

- RDF has terms for describing lists, bags, sequences, etc.
- RDF also can describe triples through reification
- Enabling statements about statements

```
:john bdi:believes _:s.
_:s rdf:type rdf:Statement.
_:s rdf:subject <http://ex.com/catalog/widgetX>.
_:s rdf:predicate cat:salePrice .
_:s rdf:object "19.95" .
```

RDF Schema (RDFS)

- **RDF Schema adds taxonomies for classes & properties**
  - subClass and subProperty
- **and some metadata.**
  - domain and range constraints on properties
- **Several widely used KB tools can import and export in RDFS**



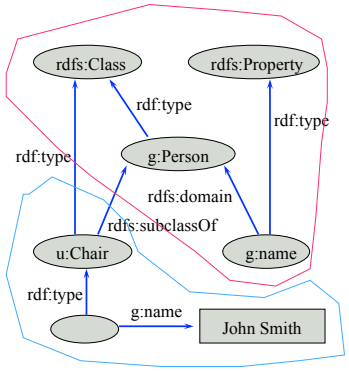
Stanford Protégé KB editor
- Java, open sourced
- extensible, lots of plug-ins
- provides reasoning & server capabilities

RDFS Vocabulary

RDFS introduces the following terms and gives each a meaning w.r.t. the rdf data model

- Terms for classes
  - rdfs:Class
  - rdfs:subClassOf
- Terms for properties
  - rdfs:domain
  - rdfs:range
  - rdfs:subPropertyOf
- Special classes
  - rdfs:Resource
  - rdfs:Literal
  - rdfs:Datatype
- Terms for collections
  - rdfs:member
  - rdfs:Container
  - rdfs:ContainerMembershipProperty
- Special properties
  - rdfs:comment
  - rdfs:seeAlso
  - rdfs:isDefinedBy
  - rdfs:label

RDF and RDF Schema




```
<rdfs:Property rdf:ID="name">
  <rdfs:domain rdf:resource="Person">
</rdfs:Property>

<rdfs:Class rdf:ID="Chair">
  <rdfs:subclassOf  rdf:resource=
    "http://schema.org/gen#Person">
</rdfs:Class>
```

```
<rdf:RDF
    xmlns:g="http://schema.org/gen"
    xmlns:u="http://schema.org/univ">
 <u:Chair rdf:ID="john">
  <g:name>John Smith</g:name>
 </u:Chair>
</rdf:RDF>
```

RDFS supports simple inferences

New and Improved! 100% Better than XML!!

- An RDF ontology plus some RDF statements may imply additional RDF statements.
- This is not true of XML.
- Note that this is **part of the data model** and not of the accessing or processing code.

```
@prefix rdfs: <http://www.....>.
@prefix : <genesis.n3>.
:parent rdfs:domain :Person;
        rdfs:range :Person.
:mother rdfs:subProperty parent;
        rdfs:domain Woman;
        rdfs:range Person
:eve :mother :cain.
```

→

```
:parent a rdf:Property.
:Person a rdf:Class.
:Woman rdfs:subClassOf Person.
:mother a rdf:Property.
:eve a :Person;
       a :Woman;
       :parent :cain.
:cain a :Person.
```

N3 example

```
@prefix rdf: <h                                  -ns#>.
@prefix rdfs: <h                                 chema#>.
@prefix : <#>
<> rdfs:comment "This is                         exa
:Person a rdfs:Class.
:Woman a rdfs:Class; rdfs:subClass
:eve a :Woman; :age "100"
:sister a rdf:Property;
       rdfs:range :Woman.
:eve :sister [a :Woman; :age 98]
:eve :believe {:eve :age "100"}.
[is :spouse of [is :sister of :eve]] :age 9
:eve.:sister.:spouse :age 99.
```

Here's how you declare a namespace.

<> Is an alias for the URI of this document.

"person is a class". The "a" syntax is sugar for rdf:type

"Woman is a class and a subclass of person". Note the ; syntax.

"eve is a woman whose age is 100."

"sister is a property from person to woman"

"eve believes that her age is 100". The braces introduce

"the spouse of the sister of eve is 99".

"the spouse of the sister of eve is 99".

Is RDF(S) better than XML?

Q: For a specific application, should I use XML or RDF?
A: It depends…

- XML's model is
  - a tree, i.e., a strong hierarchy
  - applications may rely on hierarchy position
  - relatively simple syntax and structure
  - not easy to *combine* trees
- RDF's model is
  - a *loose* collections of relations
  - applications may do database-like search
  - not easy to recover hierarchy
  - easy to combine relations in one big collection
  - great for the integration of heterogeneous information

From where will the markup come?

- A few authors will add it manually.
- More will use annotation tools.
  - SMORE: Semantic Markup, Ontology and RDF Editor
- Intelligent processors (e.g., NLP) can understand documents and add markup (hard)
  - Machine learning powered information extraction tools show promise
- Lots of web content comes from databases & we can generate SW markup along with the HTML
  - See http://ebiquity.umbc.edu/

From where will the markup come?

- In many tools, part of the metadata information is present, but thrown away at output
  - e.g., a business chart can be generated by a tool…
  - …it "knows" the structure, the classification, etc. of the chart
  - …but, usually, this information is lost
  - …storing it in metadata is easy!
- So *"semantic web aware"* tools can produce lots of metadata
  - E.g., Adobe's use of its XMP platform

Problems with RDFS

- RDFS **too weak** to describe resources in detail, e.g.
  - No *localised range and domain* constraints

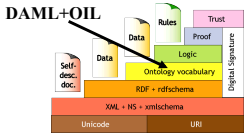
    Can't say that the range of hasChild is person when applied to persons and dog when applied to dogs
  - No *existence/cardinality* constraints

    Can't say that all *instances* of person have a mother that is also a person, or that persons have exactly 2 parents
  - No *transitive, inverse or symmetrical* properties

    Can't say isPartOf is a transitive property, hasPart is the inverse of isPartOf or touches is symmetrical
- We need RDF terms providing these and other features.

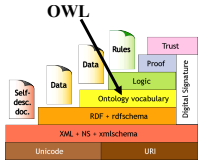
DAML+OIL = RDF + KR

- DAML = Darpa Agent Markup Language
  - DARPA program with 17 projects & an integrator developing language spec, tools, applications for SW.
- OIL = Ontology Inference Layer
  - An EU effort aimed at developing a layered approach to representing knowledge on the web.
- Process
  - Joint Committee: US DAML and EU Semantic Web Technologies participants
  - DAML+OIL specs released in 2001
  - See http://www.daml.org/
  - Includes model theoretic and axiomatic semantics



W3C' s Web Ontology Language (OWL)

- DAML+OIL begat OWL.
- OWL released as W3C recommendation 2/10/04
- See the W3C OWL pages for overview, guide, specification, test cases, etc.
- Three layers of OWL are defined of decreasing levels of complexity and expressiveness
  - **OWL Full** is the whole thing
  - **OWL DL** (Description Logic) introduces restrictions
  - **OWL Lite** is an entry level language intended to be easy to understand and implement



- Owl 2 became a W3C recommendation in 2009

OWL ↔ RDF

- An OWL ontology is a set of RDF statements
  - OWL defines semantics for certain statements
  - Does **NOT** restrict what can be said -- documents can include arbitrary RDF
  - But no OWL semantics for non-OWL statements
- Adds capabilities common to description logics, e.g., cardinality constraints, defined classes, equivalence, disjoint classes, etc.
- Support for ontologies as objects (e.g., importing, versioning, …
- But not (yet) variables, quantification, & rules
- A complete OWL reasoning is significantly more complex than a complete RDFS reasoner.

Owl is based on Description Logic

- DL is a family of KR languages that might be described as **"Logic meets Objects"**
- DL is characterized by a set of constructors that allow one to build complex **concepts** and **roles** from atomic ones
  - **Concepts** correspond to classes; interpreted as sets of objects
  - **Roles** correspond to relations; interpreted as binary relations on objects
- Axioms assert **facts** about concepts, roles & **individuals**
- Distinguished by:
  - Formal semantics for a decidable fragment of FOL
  - Sound and complete procedures for key problems
  - Many implemented systems, some highly optimized

OWL Class Constructors

Constructor	DL Syntax	Example
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	Human ⊓ Male
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	Doctor ⊔ Lawyer
complementOf	$\neg C$	¬Male
oneOf	$\{x_1 \ldots x_n\}$	{john, mary}
allValuesFrom	$\forall P.C$	∀hasChild.Doctor
someValuesFrom	$\exists P.C$	∃hasChild.Lawyer
maxCardinality	$\leqslant nP$	⩽1hasChild
minCardinality	$\geqslant nP$	⩾2hasChild

borrowed from Ian Horrocks

OWL Axioms

Axiom	DL Syntax	Example
subClassOf	$C_1 \sqsubseteq C_2$	Human $\sqsubseteq$ Animal $\sqcap$ Biped
equivalentClass	$C_1 \equiv C_2$	Man $\equiv$ Human $\sqcap$ Male
disjointWith	$C_1 \sqsubseteq \neg C_2$	Male $\sqsubseteq \neg$Female
sameIndividualAs	$\{x_1\} \equiv \{x_2\}$	{President_Bush} $\equiv$ {G_W_Bush}
differentFrom	$\{x_1\} \sqsubseteq \neg\{x_2\}$	{john} $\sqsubseteq \neg$ {peter}
subPropertyOf	$P_1 \sqsubseteq P_2$	hasDaughter $\sqsubseteq$ hasChild
equivalentProperty	$P_1 \equiv P_2$	cost $\equiv$ price
inverseOf	$P_1 \equiv P_2^-$	hasChild $\equiv$ hasParent$^-$
transitiveProperty	$P^+ \sqsubseteq P$	ancestor$^+ \sqsubseteq$ ancestor
functionalProperty	$\top \sqsubseteq \leqslant 1P$	$\top \sqsubseteq \leqslant 1$hasMother
inverseFunctionalProperty	$\top \sqsubseteq \leqslant 1P^-$	$\top \sqsubseteq \leqslant 1$hasSSN$^-$

borrowed from Ian Horrocks

OWL Language

- Three species of OWL
  - *OWL Full* is union of OWL syntax and RDF
  - *OWL DL* restricted to FOL fragment (≅ DAML+OIL)
  - *OWL Lite* is "simpler" subset of OWL DL
- Semantic layering
  - OWL DL ≅ OWL full within DL fragment
- OWL DL based on SHIQ Description Logic
- OWL DL Benefits from many years of DL research
  - Well defined semantics
  - Formal properties well understood (complexity, decidability)
  - Known reasoning algorithms
  - Implemented systems (highly optimised)

OWL Lite Features

- **RDF Schema Features**
  - *Class, rdfs:subClassOf , Individual*
  - *rdf:Property, rdfs:subPropertyOf*
  - *rdfs:domain , rdfs:range*
- **Equality and Inequality**
  - *sameClassAs , samePropertyAs , sameIndividualAs*
  - *differentIndividualFrom*
- **Restricted Cardinality**
  - *minCardinality, maxCardinality (restricted to 0 or 1)*
  - *cardinality (restricted to 0 or 1)*
- **Property Characteristics**
  - *inverseOf , TransitiveProperty , SymmetricProperty*
  - *FunctionalProperty(unique) , InverseFunctionalProperty*
  - *allValuesFrom, someValuesFrom (universal and existential local range restrictions)*
- ***Datatypes***
  - *Following the decisions of RDF Core.*
- ***Header Information***
  - *imports , Dublin Core Metadata , versionInfo*

OWL Features

- **Class Axioms**
  - *oneOf* (enumerated classes)
  - *disjointWith*
  - *sameClassAs* applied to class expressions
  - *rdfs:subClassOf* applied to class expressions
- **Boolean Combinations of Class Expressions**
  - *unionOf*
  - *intersectionOf*
  - *complementOf*
- **Arbitrary Cardinality**
  - *minCardinality*
  - *maxCardinality*
  - *cardinality*
- **Filler Information**
  - *hasValue* Descriptions can include specific value information

OWL Ontologies

- The owl:Ontology class describes an ontology
- An ontology file should be one instance of owl:Ontology
- Ontology properties include
  - owl:imports, owl:versionInfo, owl:priorVersion
  - owl:backwardCompatibleWith, owl:incompatibleWith
  - rdfs:label, rdfs:comment can also be used
- Deprecation control classes:
  - owl:DeprecatedClass, owl:DeprecatedProperty types

OWL in One Slide

OWL is built on top of XML and RDF

It allows the definition, sharing, composition and use of ontologies

OWL is ~= a frame based knowledge representation language

It can be used to add metadata about anything which has a URI.

URIs are a W3C standard generalizing URLs

everything has a URI

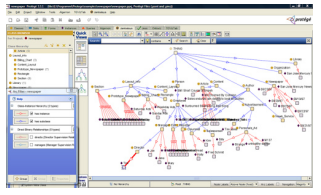
```
<rdf:RDF xmlns:rdf ="http://w3.org/22-rdf-syntax-ns#"
   xmlns:rdfs=http://w3.org/rdf-schema#>
   xmlns:owl="http://www.w3.org/2002/07/owl#">
<owl:Ontology rdf:about="">
   <owl:imports rdf:resource="http://owl.org/owl+oil"/>
</owl:Ontology>
<owl:Class rdf:ID="Person">
 <rdfs:subClassOf rdf:resource="#Animal"/>
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#hasParent"/>
   <owl:allValuesFrom rdf:resource="#Person"/>
  </owl:Restriction>
 </rdfs:subClassOf>
 <rdfs:subClassOf>
  <owl:Restriction owl:cardinality="1">
   <owl:onProperty rdf:resource="#hasFather"/>
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
<Person rdf:about="http://umbc.edu/~finin/">
 <rdfs:comment>Finin is a person.</rdfs:comment>
</Person>
```

RDFa

- RDFa is a W3C recommendation for embedding SW markup in HTML as tag attributed
- = RDF in

Ontology Editor

- There are a number of editors available for creating and editing ontologies and data
- We recommend using Protégé, a java-based free system developed at Stanford
  - Good support for reasoning
  - Lots of plugins



Triple Stores

- A triple store is a database for RDF triples
- It usually has a native API and often accepts SPARQL queries
- It might do reasoning, either in an eager manner (as triples are loaded) or on demand (to answer queries), etc
- Some stores focus on scalability and others on flexibility and features
- We'll look at several, including Sesame, Apache Jena and stardog

Frameworks and Libraries

- There are frameworks, libraries and packages for most programming languages
- RDFLib is a popular library for Python that we will study
- Jena is a very comprehensive Java framework originally developed by HP and now Apache
- Redlands is set of libraries for using RDF in C
- Others are available for Ruby, C#, Perl, PHP, Lisp, etc.