CO7216 - Semantic Web

Revision

Midsummer Examination

- **May 24th, 2016 @ 14:30 – 2 Hours (60%)**
  - Mock example paper available on the blackboard. This practice test is presented in a format that closely resembles the actual test
  - Contains 3 questions
  - Each of the questions is worth 50 marks (only the best two answers will be considered).

Examination Preparation

- Mock exam paper
- Surgeries
- Past exam papers
- Lecture Notes

Overview

- Introduction
- RDF and RDF Schema, Turtle Syntax
- Ontology Engineering
- SPARQL Language
- Web Ontology Language (OWL)
- Alternative OWL Syntax
- DL and Rule-based Reasoning
- Linked Data
- RDFa and Microdata

Overview

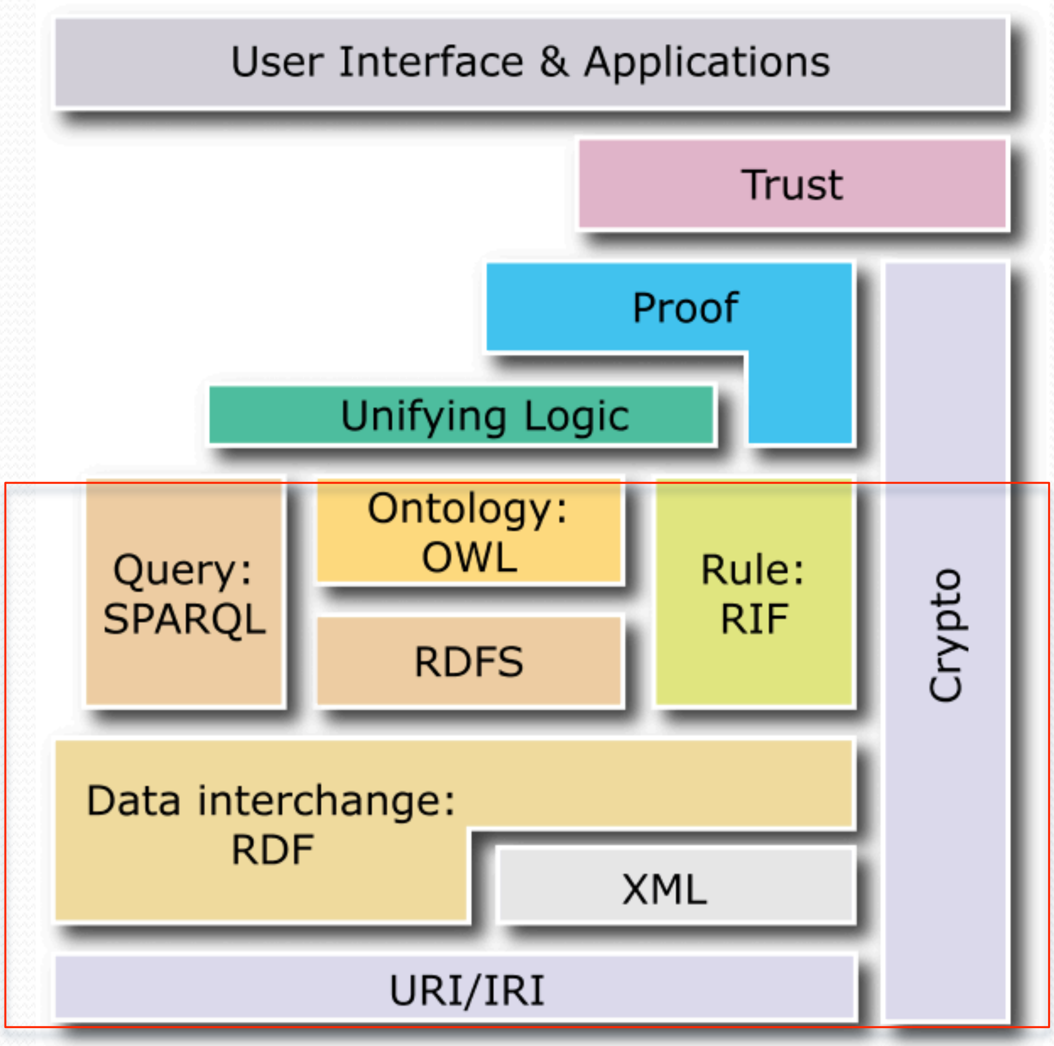
- **Introduction**
- RDF and RDF Schema, Turtle Syntax
- Ontology Engineering
- SPARQL Language
- Web Ontology Language (OWL)
- Alternative OWL Syntax
- DL and Rule-based Reasoning
- Linked Data
- RDFa and Microdata

Introduction

- Syntactic Web vs Semantic Web
- URL vs URI vs IRI
- XML, XML-S, RDF, RDFS, DAML-OIL, OWL (Expressivity)
- Open World Assumption
- Ontology life-cycle

Introduction

Semantic Web Layer Cake



Overview

- Introduction
- **RDF and RDF Schema, Turtle Syntax**
- Ontology Engineering
- SPARQL Language
- Web Ontology Language (OWL)
- Alternative OWL Syntax
- DL and Rule-based Reasoning
- Linked Data
- RDFa and Microdata

RDF Documents

- RDF Triple <subject, predicate, object>
- RDF document

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:uol="http://www.cs.le.ac.uk/rdf#">
<rdf:Description rdf:about=
        "http://www.cs.le.ac.uk/rdf#Leicester">
  <uol:population>328939</uol:population>
  <uol:isLocatedIn rdf:resource=
        "http://www.cs.le.ac.uk/rdf#Leicestershire"/>
</rdf:Description>
</rdf:RDF>
```

RDF Graph Notation

A RDF graph is a directed graph with labeled nodes and arcs

- **from** the resource (the **subject** of the statement)
- **to** the value (the **object** of the statement)

Objects could be **resources** or **literals (strings/numbers)**

RDF Schema Core Classes and Properties

- **rdfs:Resource**, the class of all resources
- **rdfs:Class**, the class of all classes
- **rdfs:Literal**, the class of all literals (string)
- **rdf:Property**, the class of all property
- **rdf:type**, relates a resource to its class
- **rdfs:subClassOf**, relates a class to one of its superclasses
- **rdfs:subPropertyOf**, relates a property to one of its superproperties
- **rdfs:domain**, specifies class of those resources that may appear as subjects in a triple with predicate P
- **rdfs:range**, specifies the class of those resources that may appear as values (of objects) in a triple with predicate P

XML vs RDF

XML

```
<?xml version="1.0"?>
<City name="Leicester">
   <county>Leicestershire</county>
   <population>328939<population>
   <postcode>LE</postcode>
  </City>
```

RDF

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:uol="http://www.cs.le.ac.uk/rdf#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xml:base="http://www.cs.le.ac.uk/rdf">
<uol:City rdf:ID="Leicester">
   <uol:isLocationIn>
     <uol:County rdf:ID="Leicestershire"/>
   </uol:isLocationIn>
   <uol:population rdf:datatype="http://www.w3.org/2001/XMLSchema#int">
    328939</uol:population>
   <uol:postcode rdf:datatype="http://www.w3.org/2001/XMLSchema#string" >
    LE</uol:postcode>
  </uol:City>
</rdf:RDF>
```

RDFS vs RDF

RDFS

```
<rdfs:Class rdf:ID="Student">
  <rdfs:subClassOf>
    <rdfs:Class rdf:ID="Person"/>
  </rdfs:subClassOf>
 </rdfs:Class>
 <rdfs:Class rdf:about="#Person">
  <rdfs:subClassOf>
    <rdfs:Class rdf:ID="LivingThing"/>
  </rdfs:subClassOf>
 </rdfs:Class>
 <rdfs:Class rdf:ID="Food"/>
 <rdfs:Class rdf:ID="Course"/>
 <rdfs:Class rdf:ID="Lecturer">
  <rdfs:subClassOf rdf:resource="#Person"/>
 </rdfs:Class>
 <rdf:Property rdf:ID="has_age">
  <rdfs:domain rdf:resource="#LivingThing"/>
  <rdfs:range rdf:resource=”&xsd;int"/>
 </rdf:Property>
 <rdf:Property rdf:ID="teach">
  <rdfs:domain rdf:resource="#Lecturer"/>
  <rdfs:range rdf:resource="#Course"/>
 </rdf:Property>
 <rdf:Property rdf:ID="eat">
  <rdfs:domain rdf:resource="#LivingThing"/>
  <rdfs:range rdf:resource="#Food"/>
 </rdf:Property>
<rdf:Property rdf:ID="study">
  <rdfs:range rdf:resource="#Course"/>
  <rdfs:domain rdf:resource="#Student"/>
 </rdf:Property>
 <rdf:Property rdf:ID="has_food_name">
  <rdfs:domain rdf:resource="#Food"/>
  <rdfs:range rdf:resource="&xsd;string"/>
 </rdf:Property>
 <rdf:Property rdf:ID="has_full_name">
  <rdfs:domain rdf:resource="#Person"/>
  <rdfs:range rdf:resource="&xsd;string"/>
 </rdf:Property>
```

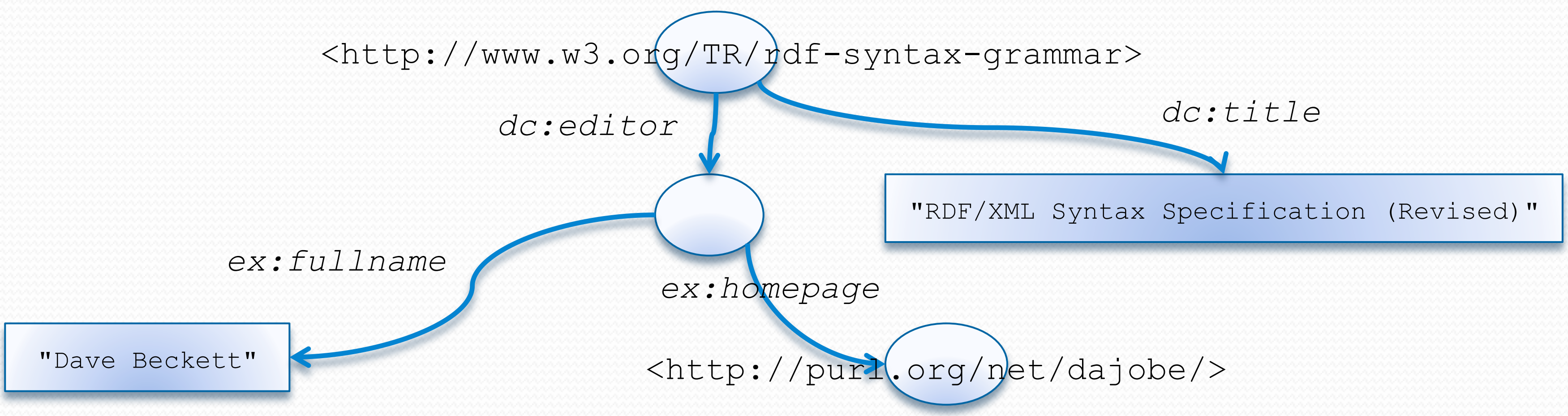
RDF

```
<uol:City rdf:ID="Leicester">
  <uol:isLocationIn>
    <uol:County rdf:ID="Leicestershire"/>
  </uol:isLocationIn>
  <uol:population rdf:datatype=“&xsd;int"
  >328939</uol:population>
  <uol:postcode rdf:datatype="&xsd;string"
  >LE</uol:postcode>
  <uol:isDistrictOf rdf:resource="#Leicestershire"/>
 </uol:City>
```

Turtle Syntax

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix ex: <http://example.org/stuff/1.0/> .

<http://www.w3.org/TR/rdf-syntax-grammar>
  dc:title "RDF/XML Syntax Specification (Revised)" ;
  dc:editor [
    ex:fullname "Dave Beckett";
    ex:homePage <http://purl.org/net/dajobe/>
  ] .
```

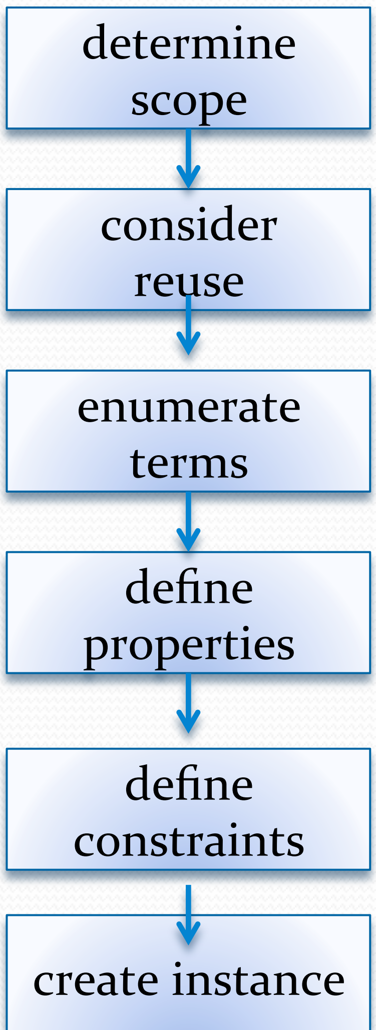


Overview

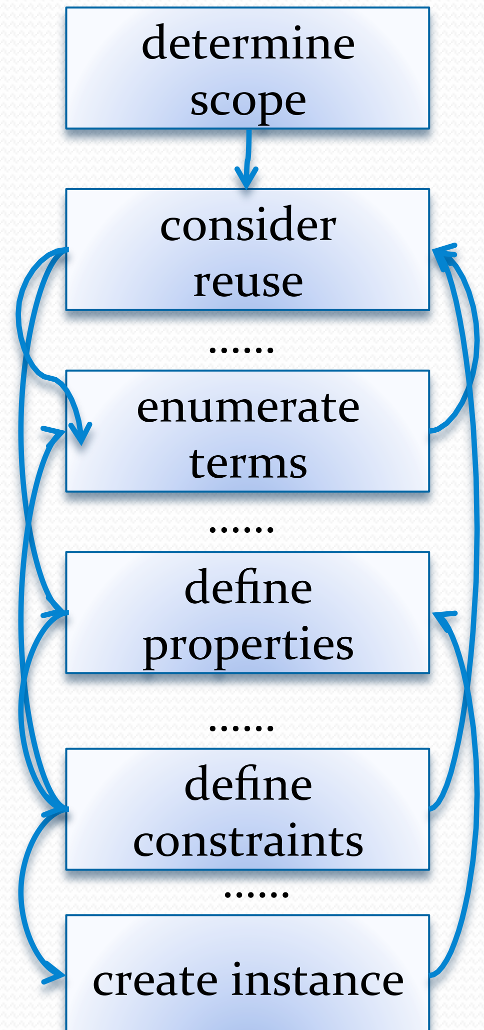
- Introduction
- RDF and RDF Schema, Turtle Syntax
- **Ontology Engineering**
- SPARQL Language
- Web Ontology Language (OWL)
- Alternative OWL Syntax
- DL and Rule-based Reasoning
- Linked Data
- RDFa and Microdata

Ontology Engineering

In this tutorial:



Reality – an iterative process



Ontology Engineering

- Class hierarchy (IS-A relationship)
- Property hierarchy
- Abox vs Tbox
- Questions to be considered:
  - Instances or classes
  - Disjoint classes
  - Class cycle
  - Scope of the Ontology

Overview

- Introduction
- RDF and RDF Schema, Turtle Syntax
- Ontology Engineering
- **SPARQL Language**
- Web Ontology Language (OWL)
- Alternative OWL Syntax
- DL and Rule-based Reasoning
- Linked Data
- RDFa and Microdata

Structure of a SELECT Query (SPARQL v1.1)

Example:

this SPARQL Query selects all persons' names

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT  ?name
FROM    <http://example.org/foaf/aliceFoaf>
WHERE   { ?x foaf:name ?name }
```

- PREFIX ..
- SELECT ..
- FROM ..
- WHERE { … }

SPARQL keywords

- `PREFIX ..`
- `SELECT ..`
- `FROM ..`
- `WHERE { … }`
- `OPTIONAL`
- `UNION`
- `FILTER`
- `ORDER BY`
- `LIMIT, OFFSET`
- `GROUP BY` (and common aggregate functions)

Examples

Query 1: Get the students taking a module taught by L1

```
PREFIX uol: <http://www.cs.le.ac.uk/rdf#>
SELECT ?student
WHERE
{
uol:L1 uol:teach ?module.
?student uol:study ?module
}
```

Query 2: Get all lecturer who are in their 30s, display their names and emails (if given)

```
PREFIX uol: <http://www.cs.le.ac.uk/rdf#>
SELECT ?p ?age ?email
WHERE {
?p a uol:Lecturer.
?p uol:has_age ?age .
OPTIONAL  {?p uol:has_email ?email}
FILTER (?age>=30 && ?age<=40)
}
```

Examples

Query 3: Show possible relationship between `s1` and `CO7216`

```
PREFIX uol: <http://www.cs.le.ac.uk/rdf#>
SELECT ?relation
WHERE {uol:s1 ?relation uol:CO7216}
```

Query 4: Count the number of students in each module

```
PREFIX uol: <http://www.cs.le.ac.uk/rdf#>
SELECT ?module  (count(DISTINCT ?s) as ?s_count)
WHERE
{
?s a uol:Student.
?s uol:study ?module.
}
GROUP BY ?module
```

Overview

- Introduction
- RDF and RDF Schema, Turtle Syntax
- Ontology Engineering
- SPARQL Language
- **Web Ontology Language (OWL)**
- Alternative OWL Syntax
- DL and Rule-based Reasoning
- Linked Data
- RDFa and Microdata

RDF Schema vs OWL

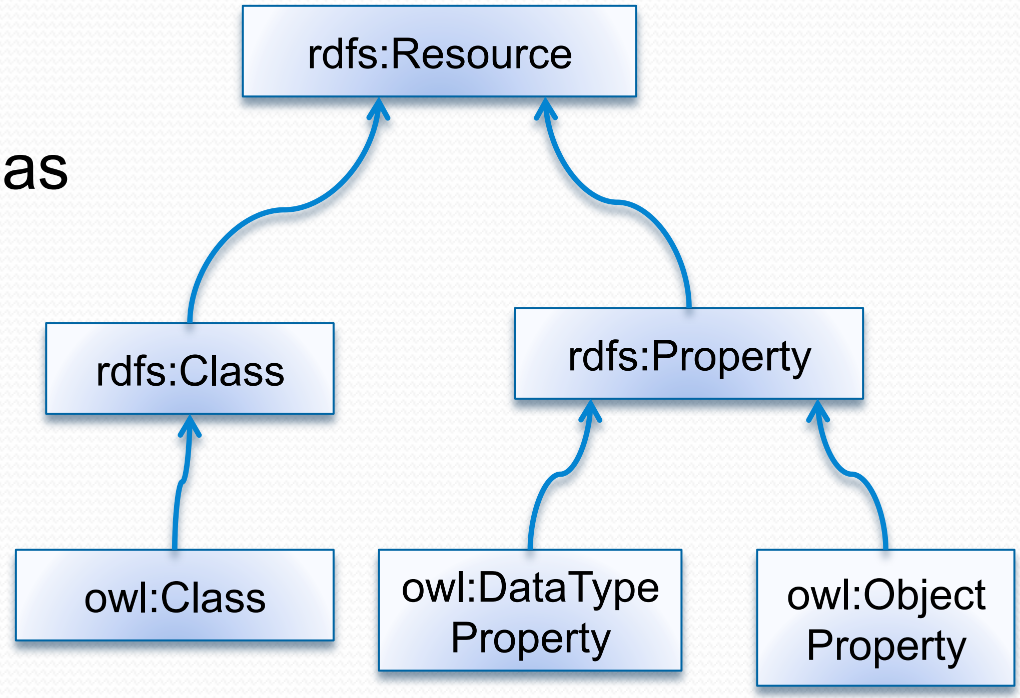
- Local scope of properties
- Disjointness of classes
- Combinations of classes
- Cardinality restrictions
- Special characteristics of properties
- …

OWL1 and OWL 2 Profiles

- OWL is developed as a vocabulary extension of RDF (the Resource Description Framework)
- Specifications
  - OWL: W3C Recommendation (2004)
    - Sublanguages: OWL Full, OWL DL, OWL lite
  - OWL2: W3C Recommendation (2009)
    - OWL 2 EL: used for ontologies containing a large numbers of properties and/or classes
    - OWL 2 QL: used for ontology-based applications that have large volumes of instance data, and where query answering is the most important reasoning task.
    - OWL2 RL: used for applications that require scalable reasoning without sacrificing too much expressive power

OWL Compatibility with RDF Schema

- All varieties of OWL use RDF for their syntax
- Instances are declared as in RDF, i.e. using RDF descriptions
- and typing information and OWL constructors are specialisations of their RDF counterparts



OWL property characteristics

- Symmetric property.
- Transitive property.
- Functional property.
- Reflexive Property
- Irreflexive Property
- Asymmetric Property
- EquivalentProperty
- DisjointProperty
- Inverse Functional property.
- inverseOf

OWL class and property restrictions

- owl:Restriction
- owl:allValuesFrom
- owl:someValuesFrom
- owl:hasValue
- owl:cardinality, minCardinality, maxCardinality
- owl:equivalentTo

Constructing OWL class using set operator

- owl:intersectionOf
- owl:unionOf
- owl:complementOf

- owl:one of
- owl:equivalentClass
- owl:disjointWith
- owl:sameIndividualAs
- owl:differentFrom

Overview

- Introduction
- RDF and RDF Schema, Turtle Syntax
- Ontology Engineering
- SPARQL Language
- Web Ontology Language (OWL)
- **Alternative OWL Syntax**
- DL and Rule-based Reasoning
- Linked Data
- RDFa and Microdata

Alternative OWL Syntax

- RDF/XML Syntax
- OWL Functional Syntax
- OWL Manchester Syntax

OWL Functional Syntax

- Follows the structural specification and allows OWL 2 ontologies to be written as text
- More compact and human-readable than RDF/XML

```
#Classes
Declaration(Class(:Person))
Declaration(Class(:Lecturer))
SubClassOf(:Lecturer :Person)
Declaration(Class(:Course))

#Properties
Declaration(ObjectProperty(:teach))
ObjectPropertyDomain(:teach :Lecturer)
ObjectPropertyRange(:teach :Course)
```

OWL Functional Syntax (2)

- ClassAssertion and PropertyAssertion

```
#ClassAssertion and PropertyAssertion

#Individual
Declaration(NamedIndividual(:CO1003))
ClassAssertion(:Course  :CO1003)
ClassAssertion(:Student :S1)

#PropertyAssertation
DataPropertyAssertion
 (:course_name uol:CO1003 "Program Design"^^xsd:string)
ObjectPropertyAssertion(:has_registered_student :CO1003 :S1)
```

OWL Functional Syntax (3)

- ClassAssertion and PropertyAssertion

```
# a module can have at most 100 registered students

SubClassOf(:Course owl:Thing)
SubClassOf(:Course
     ObjectMaxCardinality(100 :has_registered_student))

# an interesting module is a module taught by a good lecturer

EquivalentClasses(:InterestingCourse
ObjectIntersectionOf(:Course
   ObjectSomeValuesFrom(: taught_by:GoodLecturer)))


DifferentIndividuals(:CO1001 :CO1003 :CO1005 :CO1007
:CO1012 :CO1016))
```

OWL Manchester Syntax

Syntax:

Class: :classID

SubClassOf:
ClassExpression
…
EquivalentTo:
ClassExpression
…
DisjointWith:
ClassExpression
…

An Example:

Class: :VegetarianPizza

SubClassOf:
owl:Thing

EquivalentTo:

Pizza **AND**
NOT (:hasTopping **some** :FishTopping)
NOT (:hasTopping **some** :MeatTopping)

DisjointWith:
:nonVegetarianPizza

OWL Manchester Syntax

- Follows the structural specification and allows OWL 2 ontologies to be written in a compact form

```
Individual: :CO1003

    Types:
        :Course

    Facts:
     :has_registered_student  :S1,
     :is_taught_by  uol:L1,
     :course_id  "CO1003"^^xsd:string,
     :course_name  "Program Design"
      ^^xsd:string

Class: :InterestingCourse

    EquivalentTo:
        :Course
         and (:is_taught_by some :GoodLecturer)
```

```
DifferentIndividuals:

:CO1001,:CO1003,:CO1005,:CO1007,
:CO1012,:CO1016
```

Overview

- Introduction
- RDF and RDF Schema, Turtle Syntax
- Ontology Engineering
- SPARQL Language
- Web Ontology Language (OWL)
- Alternative OWL Syntax
- **DL and Rule-based Reasoning**
- Linked Data
- RDFa and Microdata

Reasoners

- OWL DL reasoning
  - Consistency
  - Subsumption
  - Satisfiability
  - Instantiation
- Rule-based Reasoning
  - SWRL Rules
  - Jena Rules

Semantic Web Rule Language - SWRL

- In SWRL syntax, a rule has the form:

  Antecedent -> Consequent

  - both antecedent and consequent are conjunctions of atoms written a1 ^ a2 ... ^ an. Variables are indicated using the standard convention of prefixing them with a question mark.
  - e.g. hasParent(?x,Tom) represents a triple statement

    Variable ?x hasParent Tom

- For example:
  - *hasParent(?x,?y) ^ hasBrother(?y,?z) ->hasUncle(?x,?z)*
  - *hasParent(?x,?y) ^ Man(?y)->Father(?y)*
  - *hasChild(?x,?y) ^ Man(?y)-> hasSon(?x,?y)*
  - *hasParent(?x,?y) ^ hasParent(?z,?y) ^ Man(?z)-> hasBrother(?x,?z)*
  - *hasSibling(?x,?y) ^ Man(?y)->hasBrother(?x,?y)*

  ..

Jena Reasoning Rules

- Jena Reasoning rules are similar to SWRL rules but its predicate can also be a variable.
- Jena Reasoning Rule Syntax:

  Antecedent -> Consequent

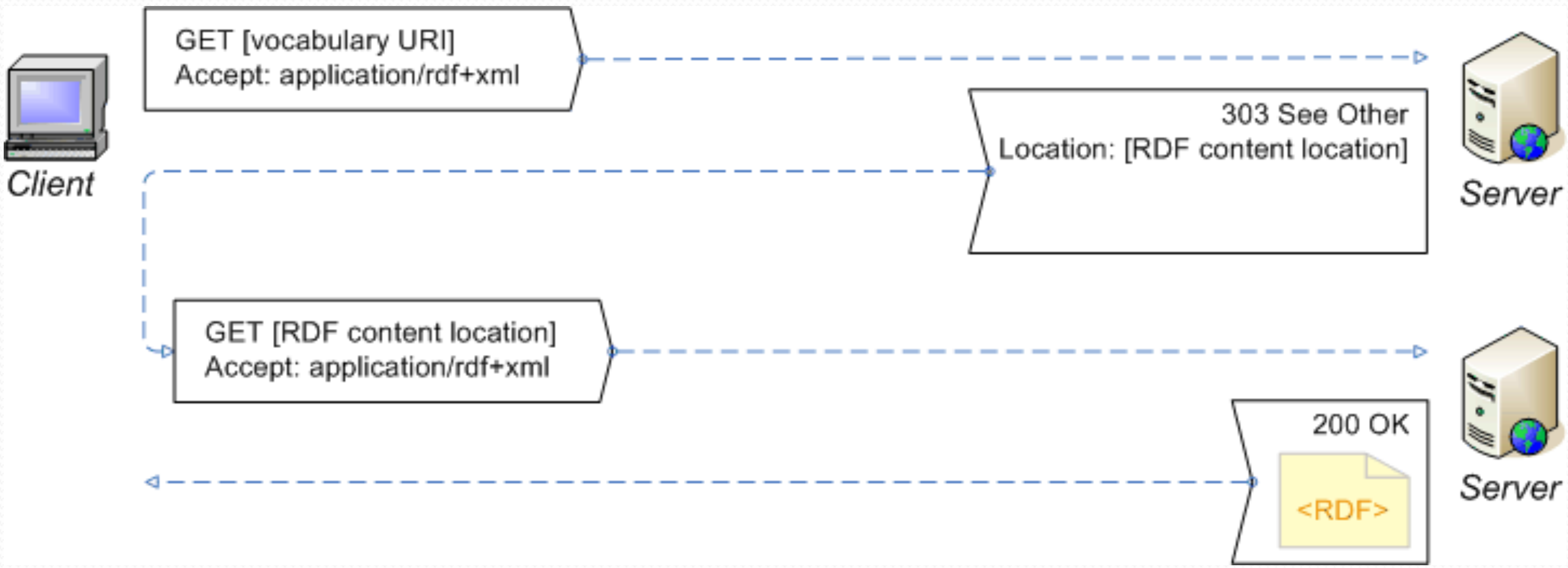
  - both antecedent and consequent are conjunctions of triple pattern written t1, t2 ... , tn.
- Examples:
  - [rule1: (?a rdfs:subClassOf ?b), (?b rdfs:subClassOf ?c)->(?a rdfs:subClassOf ?c)]
  - [transitive_rule1: (?p rdf:type owl:TransitiveProperty),(?a ?p ?b), (?b ?p ?c)-> (?a ?p ?c)]

Overview

- Introduction
- RDF and RDF Schema, Turtle Syntax
- Ontology Engineering
- SPARQL Language
- Web Ontology Language (OWL)
- Alternative OWL Syntax
- DL and Rule-based Reasoning
- **Linked Data**
- RDFa and Microdata

Setup Your Infrastructure

- Content Negotiation (303 Redirect)



Linked Data

- Web of Linked Documents vs Web of Linked Data
- Making URIs Defererenceable
  - 303 redirect vs Hash URIs
- Content Negotiation
- Steps to Publishing Linked Data
  - Understand the Principles
  - Understand your Data
  - Choose URIs for Things in your Data
  - Setup Your Infrastructure
  - Link to other Data Sets

Overview

- Introduction
- RDF and RDF Schema, Turtle Syntax
- Ontology Engineering
- SPARQL Language
- Web Ontology Language (OWL)
- Alternative OWL Syntax
- DL and Rule-based Reasoning
- Linked Data
- **RDFa and Microdata**

RDFa

- Annotating HTML document using RDFa
  - property
  - prefix
  - vocab
  - resource
  - typeof
  - rel
- Annotating HTML document Microdata
  - Itemscope
  - itemprop
  - Itemtype
  - itemid
  - itemref