MANCHESTER 1824

The University of Manchester

An Introduction to RDF(S) and a Quick Tour of OWL

MANCHESTER 1824

The University of Manchester

Ontology

- Borrowed from philosophy - the study of “The nature of being”
- “A specification of a conceptualisation” [Gruber]

Ontology

- Borrowed from philosophy - the study of “The nature of being”
- “A specification of a conceptualisation” [Gruber]
- In general, an ontology provides a mechanism to capture information about the objects and the relationships that hold between them in some domain of interest.

Ontology Languages

- Wide variety of ontology languages - some more formal than others.
  - Semantic Networks
  - Topic Maps
  - UML
  - RDF
  - OWL

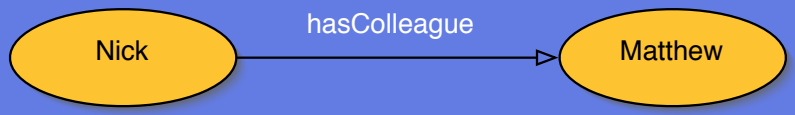
MANCHESTER 1824

The University of Manchester

RDF - Resource Description Framework

- RDF is a graphical language used for representing information about resources on the web. It is a basic ontology language.
- Resources are described in terms of properties and property values using RDF statements.
- Statements are represented as triples, consisting of a subject, predicate and object. [S, P, O]

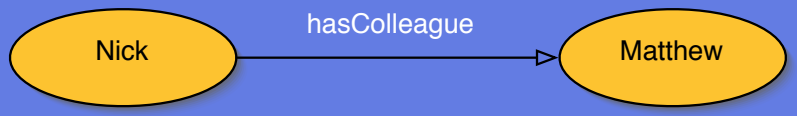
RDF Example



MANCHESTER 1824

The University of Manchester

RDF Example



- Subject: Nick
- Predicate: hasColleague
- Object: Matthew

```
<Nick> <hasColleague> <Matthew>.
```

MANCHESTER 1824

The University of Manchester

Naming Resources In RDF

- RDF uses URIs - Unique Resource Identifiers to identify resources.

Naming Resources In RDF

- RDF uses URIs - Unique Resource Identifiers to identify resources.
- Actually, to be more precise RDF uses URIRefs to identify resources.
  - A URIRef consists of a URI and an optional Fragment Identifier separated from the URI by the hash symbol #. For example,

```
http://www.co-ode.org/people#hasColleague
```

MANCHESTER 1824

The University of Manchester

Naming Resources In RDF

- RDF uses URIs - Unique Resource Identifiers to identify resources.
- Actually, to be more precise RDF uses URIRefs to identify resources.
  - A URIRef consists of a URI and an optional Fragment Identifier separated from the URI by the hash symbol #. For example,

```
http://www.co-ode.org/people#hasColleague
```

```
coode:hasColleague
```

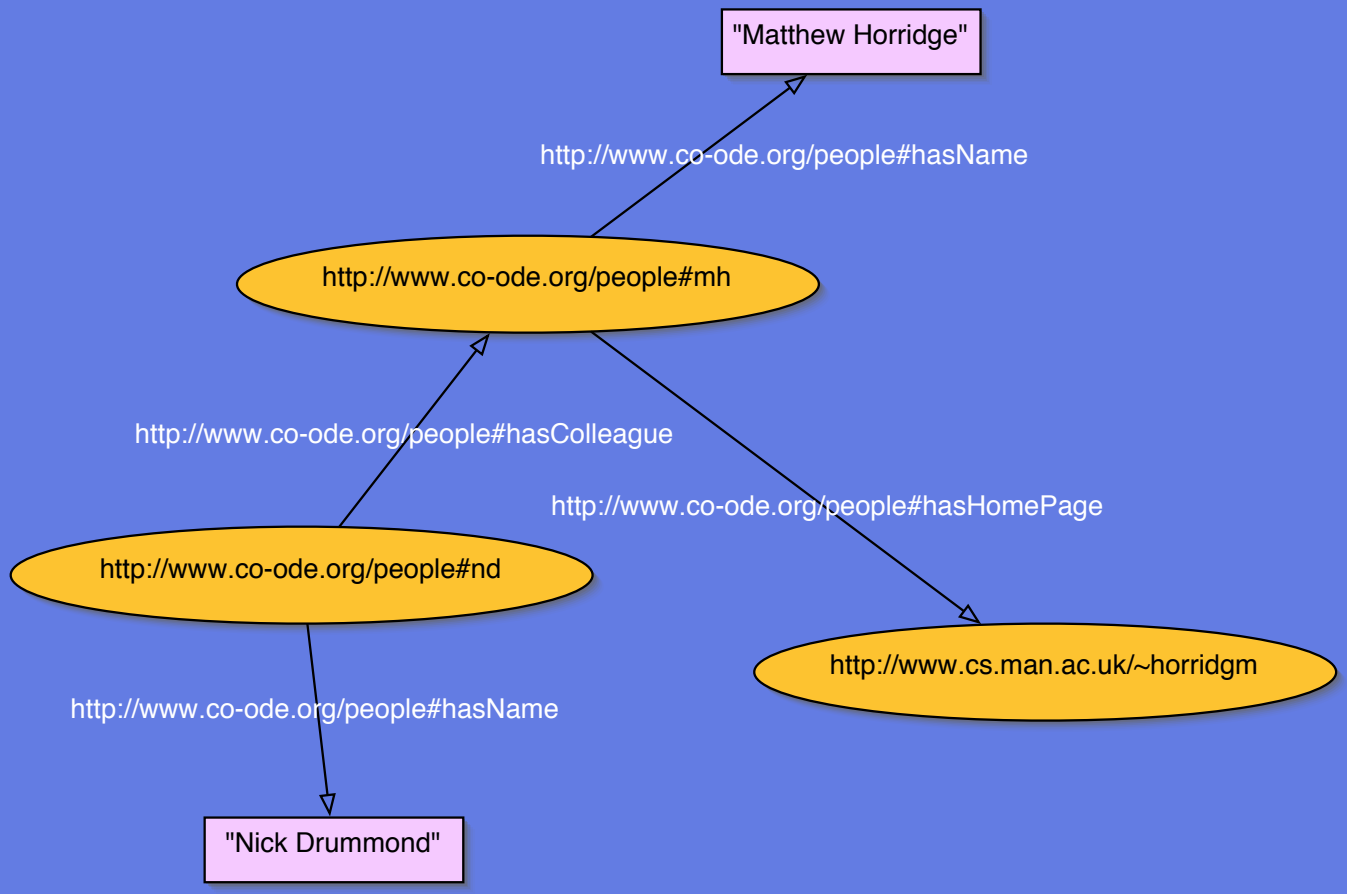
Vocabularies

- A set of URIRefs is known as a vocabulary
  - The RDF Vocabulary - The set of URIRefs used in descibing the RDF concepts e.g. `rdf:Property`, `rdf:Resource`, `rdf:type`.
  - The RDFS Vocabulary - The set of URIRefs used in describing the RDF Schema langauge e.g. `rdfs:Class`, `rdfs:domain`
  - The 'Pizza Ontology' Vocabulary - `pz:hasTopping`, `pz:Pizza`, `pz:VegetarianPizza`

Linking Statements

- The subject of one statement may be the object of another statement.
- A set of linked statements (triples) forms an RDF Graph.

An RDF Graph Example



RDF Serialisation
RDF/XML

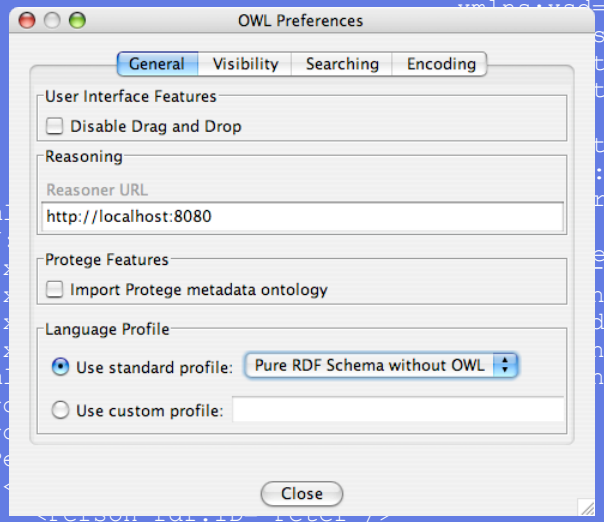
```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
      xmlns:coode="http://www.co-ode.org/people#"
      xml:base="http://www.co-ode.org/people">
<rdf:Description rdf:ID="mh">
         <coode:hasHomepage rdf:resource="http://www.cs.man.ac.uk/~horridgm"/>
         <coode:hasName>Matthew Horridge</coode:hasName>
</rdf:Description>
<rdf:Description rdf:ID="nd">
         <coode:hasName>Nick Drummond</coode:hasName>
         <coode:hasColleage rdf:resource="#mh"/>
</rdf:Description>
</rdf:RDF>
```

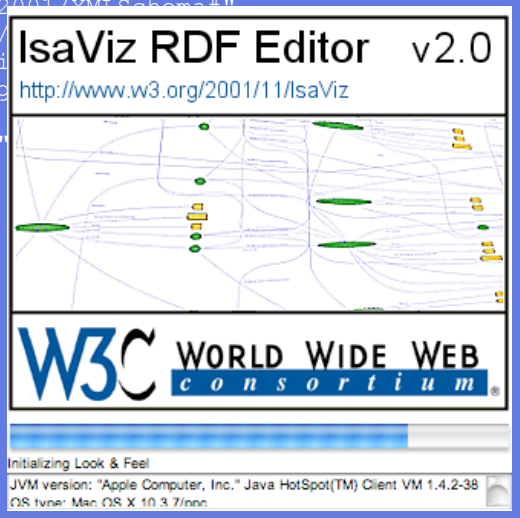
```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:coode="http://www.co-ode.org/people#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xml:base="file:/Users/matthewhorridge/Desktop/Test.rdf">
  <rdf:Description rdf:about="http://www.co-ode.org/people#nd">
    <coode:hasName>Nick Drummond</coode:hasName>
    <coode:hasColleage>
      <rdf:Description rdf:about="http://www.co-ode.org/people#mh">
        <coode:hasName>Matthew Horridge</coode:hasName>
        <coode:hasHomepage rdf:resource="http://www.cs.man.ac.uk/~horridgm"/>
      </rdf:Description>
    </coode:hasColleage>
  </rdf:Description>
</rdf:RDF>
```

Editing RDF(S)

- IsaViz RDF Editor
- Protege-OWL in RDF(S) mode

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns="http://www.w3.org/
    xml:base="http://www.owl-ontologi
    xmlns:owl="http://www.owl-ontolog
    <owl:Class rdf:ID="Person"/>
    <owl:ObjectProperty rdf:ID="hasFather"
    <Person rdf:ID="Matthew">
      <hasFather>
        <Person rdf:ID="Peter"/>
      </hasFather>
    </Person>
</rdf:RDF>
```





MANCHESTER 1824

The University of Manchester

RDF Schema (RDFS) - The RDF Vocabulary Description Language

- RDF Schema 'semantically extends' RDF to enable us to talk about classes of resources, and the properties that will be used with them.
- It does this by giving special meaning to certain rdf properties and resources.
- RDF Schema provides the means to describe application specific RDF vocabularies.

Describing Classes with RDFS

- To describe classes we can use built in RDF Schema resources:

- `rdfs:Class`

- `rdfs:subClassOf`

- These are used in conjunction with the `rdf:type` property.

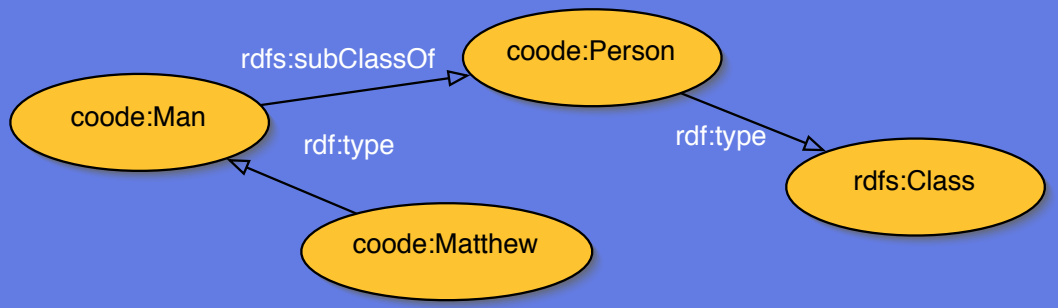
Describing Classes with RDFS

- To describe classes we can use built in RDF Schema resources:

- `rdfs:Class`

- `rdfs:subClassOf`

- These are used in conjunction with the `rdf:type` property.



MANCHESTER 1824

The University of Manchester

Describing Properties with RDF(S)

- RDF Schema allows us to describe properties. (Properties are instances of the class **`rdf:Property`**!)

- We can specify a domain using **`rdfs:domain`**.

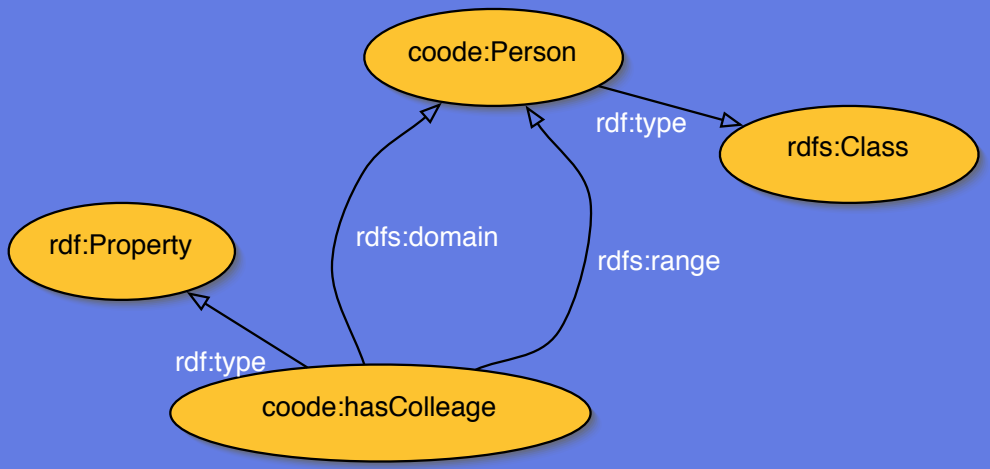
- We can specify a range using **`rdfs:range`**.

Describing Properties with RDF(S)

- RDF Schema allows us to describe properties.
  (Properties are instances of the class **`rdf:Property`**!)

- We can specify a domain using **`rdfs:domain`**.

- We can specify a range using **`rdfs:range`**.



Other RDFS Built-In Properties

- `rdfs:subPropertyOf`
- `rdfs:comment`
- `rdfs:label`
- `rdfs:seeAlso`
  - `rdfs:isDefinedBy`

MANCHESTER 1824

The University of Manchester

RDF(S) Summary

- RDF - The Resource Description Framework allows us to describe resources by specifying their properties and property values.
- RDF Statements are triples of the form [Subject, Predicate, Object]
- A set of RDF triples forms an RDF Graph
- RDF Schema semantically extends RDF by providing a means to describe RDF Vocabularies.

MANCHESTER 1824

The University of Manchester

RDF(S) Summary

- RDF and RDF Schema provide basic capabilities for describing vocabularies that describe resources.

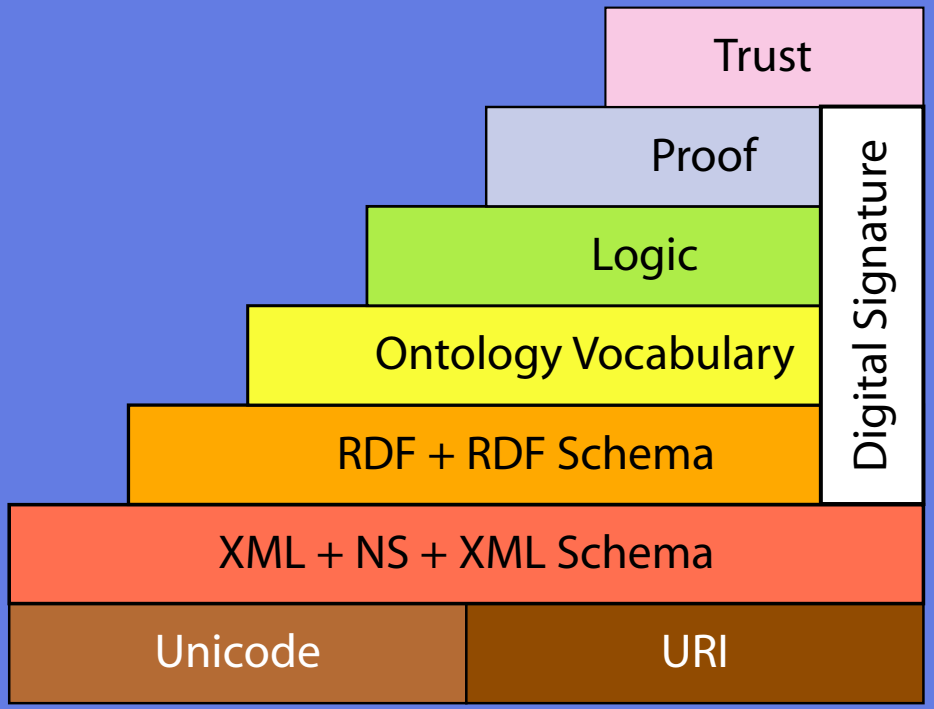

RDF(S) Summary

- RDF and RDF Schema provide basic capabilities for describing vocabularies that describe resources.
- However, certain other capabilities are desirable e.g.:
  - Cardinality constraints, specifying that properties are transitive, specifying inverse properties, specifying the 'local' range and/or cardinality for property when used with a given class, the ability to describe new classes by combining existing classes (using intersections and unions), negation (using 'not').

OWL

- Latest standard in ontology languages from the World Wide Web Consortium (W3C).
- Built on top of RDF (OWL semantically extends RDF(S)), and based on its predecessor language DAML+OIL.
- OWL has a rich set of modelling constructors.
- Three 'species': OWL-Lite, OWL-DL and OWL-Full.

The "Layer Cake"



The Three Species of OWL

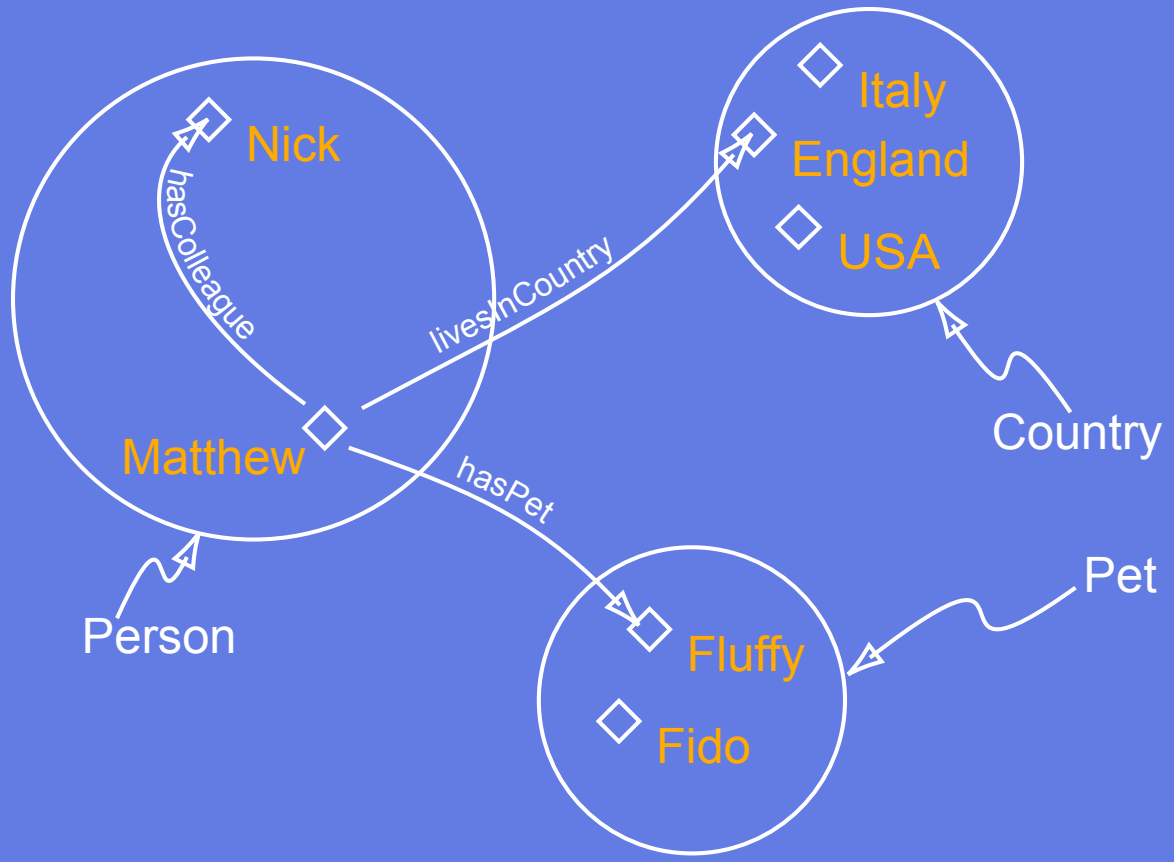
- OWL-Full - No restrictions on how/where language constructs can be used. The union of OWL and RDF(S). OWL-Full is not decidable.

- OWL-DL - Restricted version of OWL-Full. Corresponds to a description logic. Certain restrictions on how/where language constructs can be used in order to guarantee decidability.

- OWL-Lite - A subset of OWL-DL. The simplest and easiest to implement of the three species.

Components of an OWL Ontology

- Individuals
- Properties
- Classes



MANCHESTER 1824

The University of Manchester

Reasoning

- For ontologies that fall into the scope of OWL-DL, we can use a reasoner to infer information that isn't explicitly represented in an ontology. Standard 'reasoning services' are:
  - Subsumption testing
  - Equivalence testing
  - Consistency testing
  - Instantiation testing

MANCHESTER 1824

The University of Manchester

Named Classes

- OWL is an ontology language that is primarily designed to describe and define classes. Classes are therefore the basic building blocks of an OWL ontology.

- OWL supports six main ways of describing classes - The simplest of these is a Named Class. The other types are: Intersection classes, Union classes, Complement classes, Restrictions, Enumerated classes.

Intersection Classes

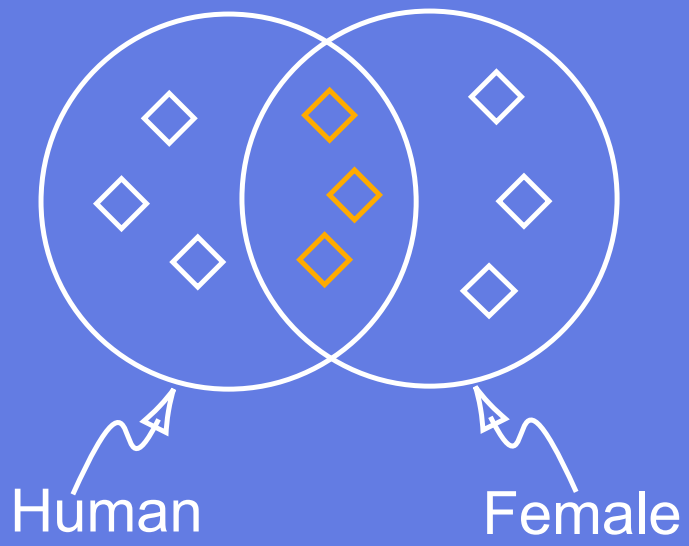
- Intersection Classes are formed by combining two or more classes with the intersection (AND) operator.

MANCHESTER 1824

The University of Manchester

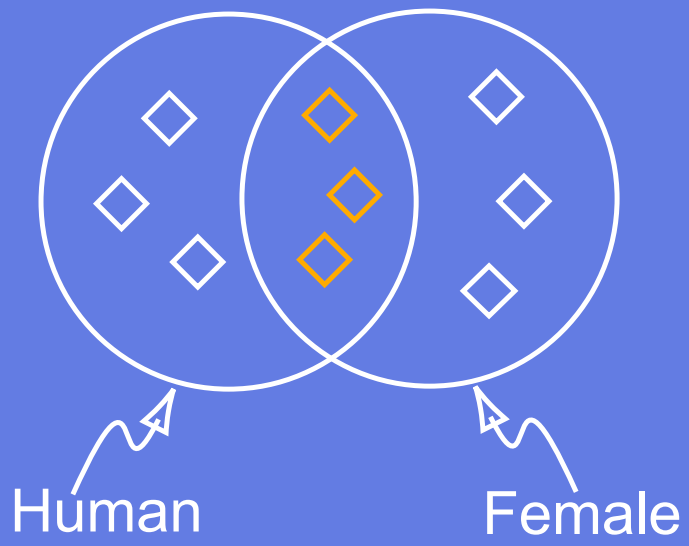
Intersection Classes

- Intersection Classes are formed by combining two or more classes with the intersection (AND) operator.



Intersection Classes

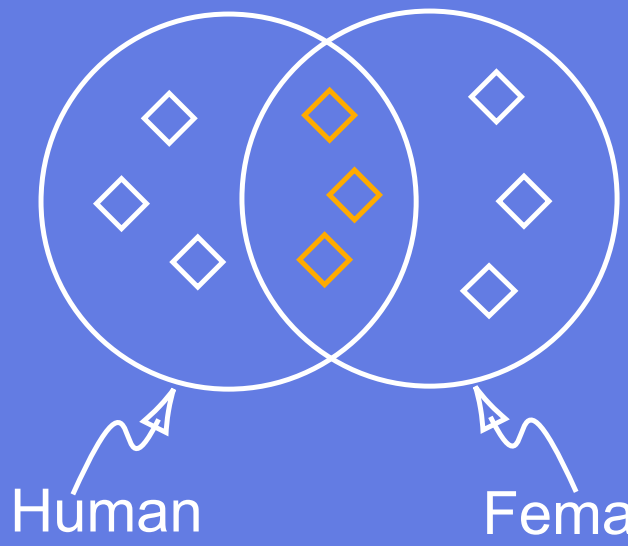
- Intersection Classes are formed by combining two or more classes with the intersection (AND) operator.



In description logics (and in Protege-OWL) we use the intersection symbol ⊓)

Intersection Classes

- Intersection Classes are formed by combining two or more classes with the intersection (AND) operator.



In description logics (and in Protege-OWL) we use the intersection symbol ⊓)

Human ⊓ Female

MANCHESTER 1824

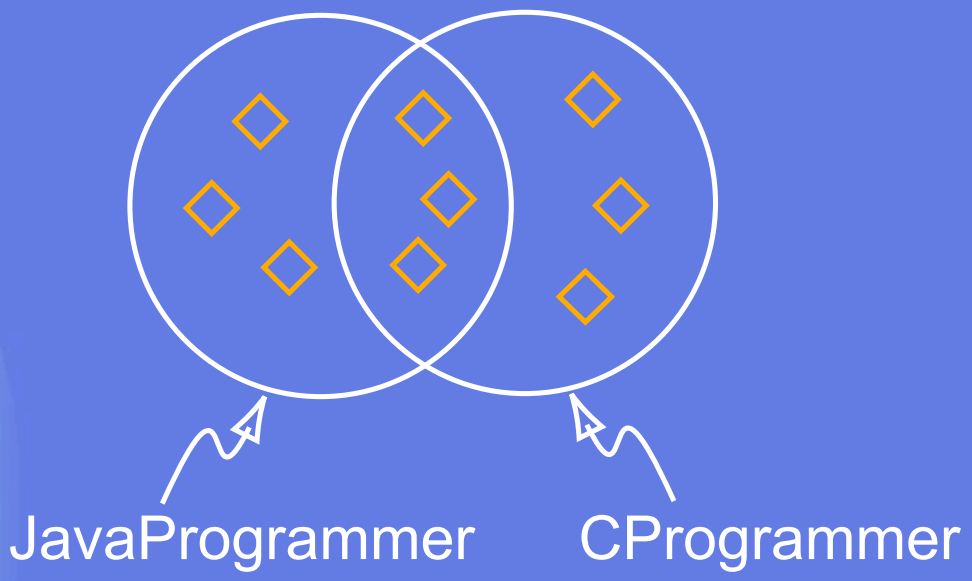
The University of Manchester

Union Classes

- Union Classes are formed using the union (OR) operator with two or more classes.

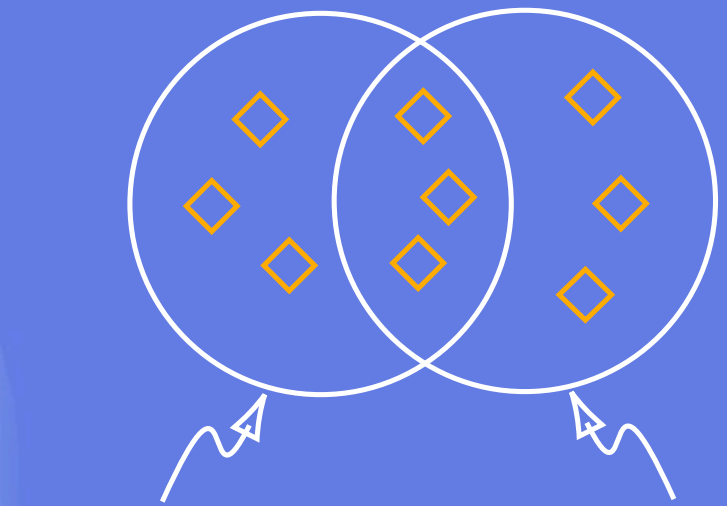
Union Classes

- Union Classes are formed using the union (OR) operator with two or more classes.



Union Classes

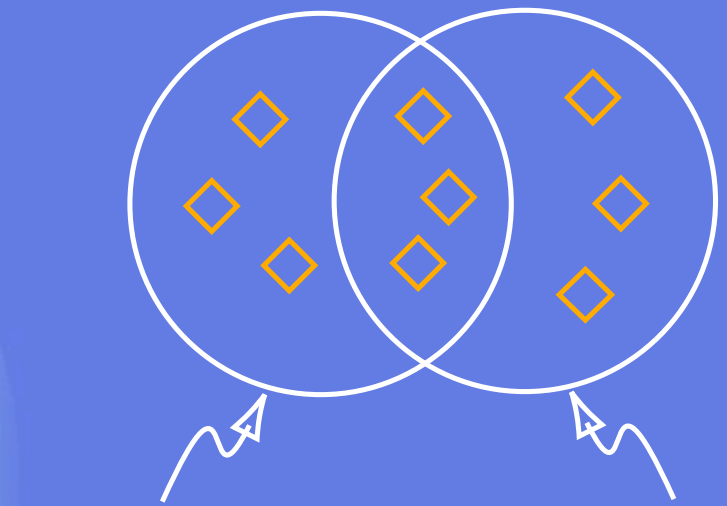
- Union Classes are formed using the union (OR) operator with two or more classes.



In description logics (and in Protege-OWL) we use the union symbol ⊔

Union Classes

- Union Classes are formed using the union (OR) operator with two or more classes.



JavaProgrammer CProgrammer

In description logics (and in Protege-OWL) we use the union symbol ⊔

JavaProgrammer ⊔ CProgrammer

MANCHESTER 1824

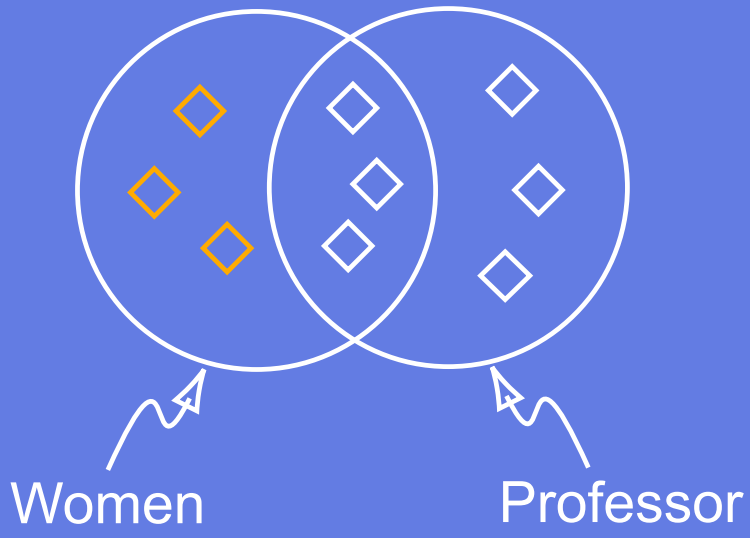
The University of Manchester

Complement Classes

- A complement class is specified by negating another class. It will contain the individuals that are not in the negated class.

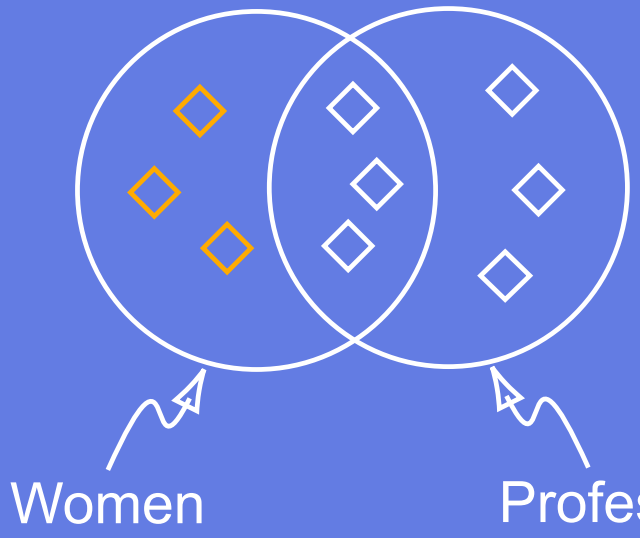

Complement Classes

- A complement class is specified by negating another class. It will contain the individuals that are not in the negated class.



Complement Classes

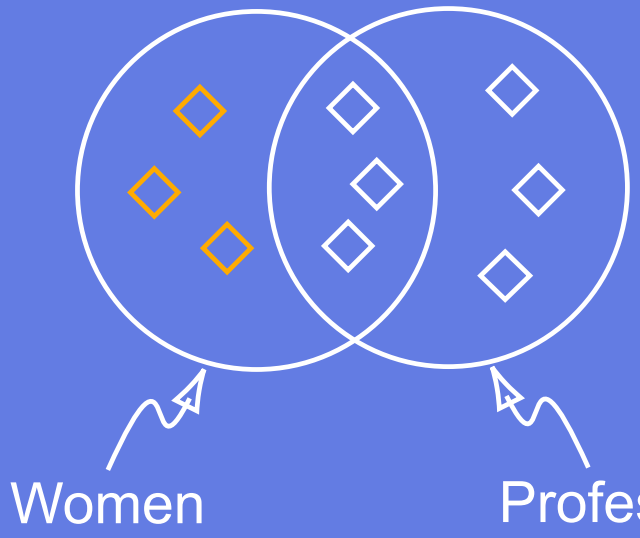
- A complement class is specified by negating another class. It will contain the individuals that are not in the negated class.



In Description Logics (and in Protege-OWL) the negation symbol ¬ is used.

Complement Classes

- A complement class is specified by negating another class. It will contain the individuals that are not in the negated class.



In Description Logics (and in Protege-OWL) the negation symbol ¬ is used.

¬ Professor ⊓ Woman

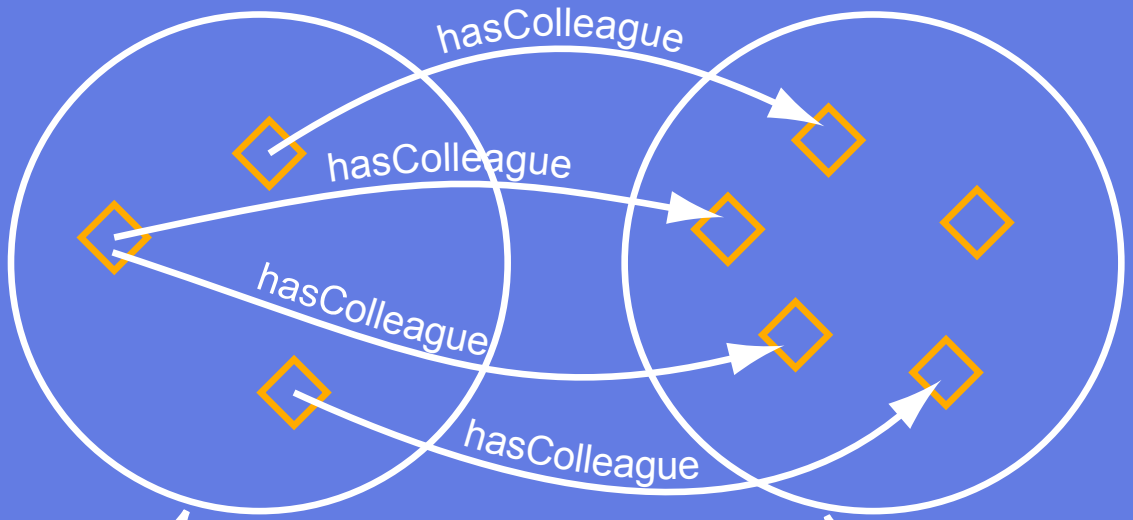
Restrictions

- Restrictions describe a class of individuals based on the type and possibly number of relationships that they participate in.
- Restrictions can be grouped into three main categories:
  - Quantifier Restrictions (Existential ∃, Universal ∀)
  - Cardinality Restrictions (Min ≤, Equal =, Max ≥)
  - Has Value Restriction (∋)

Existential Restrictions

- The most common type of restriction we will use is an existential restriction, which has the symbol ∃ (backwards E).
- The existential restriction means 'some values from', or at least one.
- An existential restriction describes the class of individuals that have at least one kind of relationship along a specified property to an individual that is a member of a specified class.

Existential Restrictions



∃ hasColleague Lecturer

Lecturer

Other Restrictions

- Universal - ∀ (upside down A) 'all values from', or only. For a given property, all the individuals must be members of a specified class.

- Cardinality Restrictions - For a given property, cardinality restrictions allow us to talk about the number of relationships that a class of individuals participate in.

- Has Value Restrictions - Allow us to specify that class of individuals that participate in a specified relationship with a specific individual.

Enumeration Classes

- An enumeration class is specified by explicitly and exhaustively listing the individuals that are members of the enumeration class.

HolidayDestinations

To specify an enumerated class, the individuals that are members of the class are listed inside curly brackets {...}

{Spain Germany France Italy}

Properties

- There are two main categories of properties: Object properties and datatype properties.
- Object properties link individuals to individuals.
- Datatype properties link individuals to datatype values (e.g. integers, floats, strings).
- Object properties my have an inverse property e.g. the inverse of worksFor might be employs.
- Properties can have as specified domain and range.

Property Characteristics

- We can specify certain property characteristics.
  - Functional - For a given individual, the property takes only one value.
  - Inverse functional - The inverse of the property is functional.
  - Symmetric - If a property links A to B then it can be inferred that it links B to A.
  - Transitive - If a property links A to B and B to C then it can be inferred that it links A to C.

MANCHESTER 1824

The University of Manchester

OWL Summary

- OWL is the latest standard in ontology languages.
- It is layered on top of RDF and RDFS, and has a rich set of constructs.
- There are three species of OWL: OWL-Lite, OWL-DL and OWL-Full.
- We can perform automated reasoning over ontologies that are written in OWL-Lite and OWL-DL.

MANCHESTER 1824

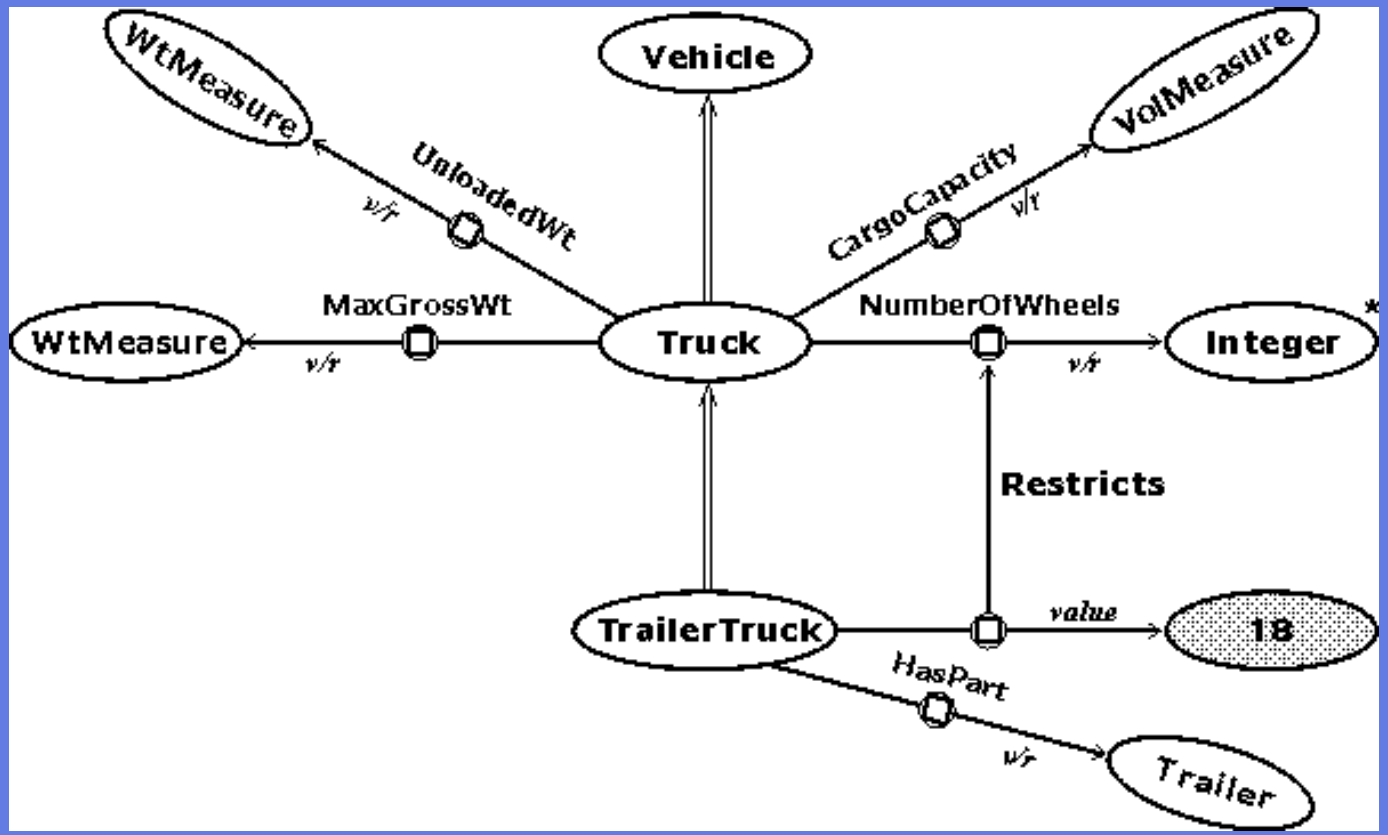
The University of Manchester

More Information

- W3C OWL Web Site
  - **http://www.w3.org/2004/OWL/**
- CO-ODE Web Site
  - **http://www.co-ode.org**
- Protege-OWL Web Site
  - **http://protege.stanford.edu/plugins/owl**

The University of Manchester

Semantic Network Example



Conceptual Graphs Example