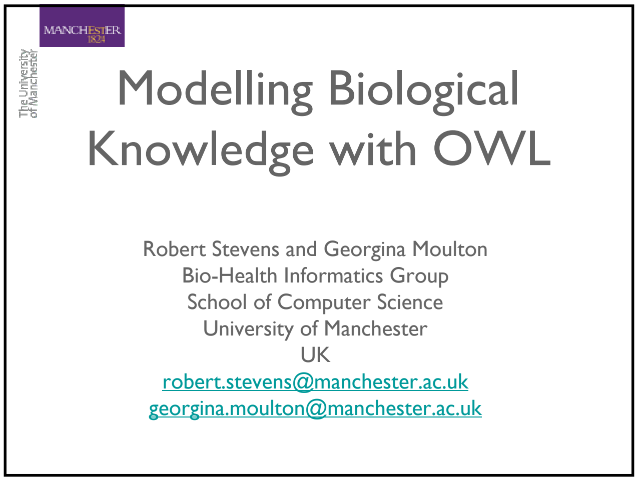
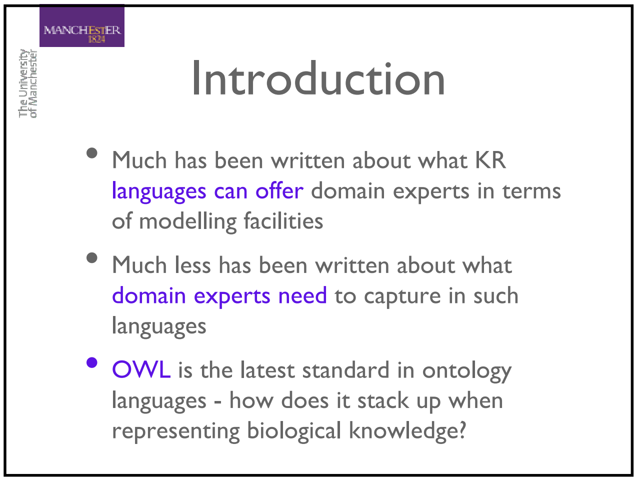


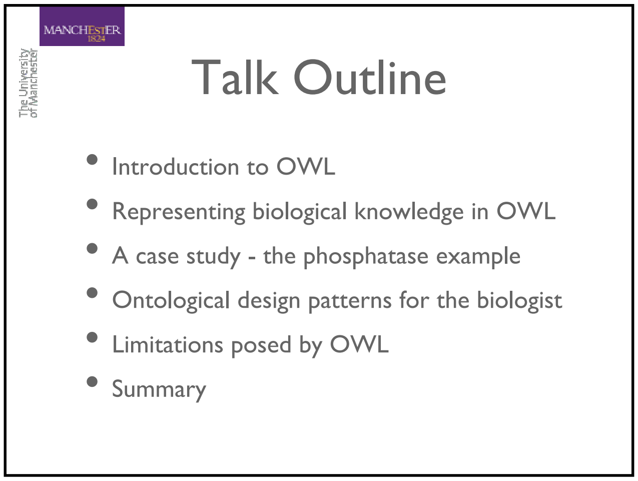
Modelling Biological Knowledge with OWL

Robert Stevens and Georgina Moulton
Bio-Health Informatics Group
School of Computer Science
University of Manchester
UK
robert.stevens@manchester.ac.uk
georgina.moulton@manchester.ac.uk



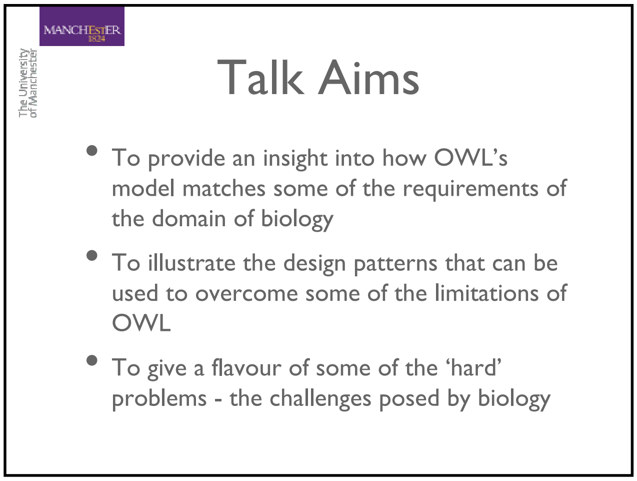
Introduction

- Much has been written about what KR languages can offer domain experts in terms of modelling facilities
- Much less has been written about what domain experts need to capture in such languages
- OWL is the latest standard in ontology languages - how does it stack up when representing biological knowledge?



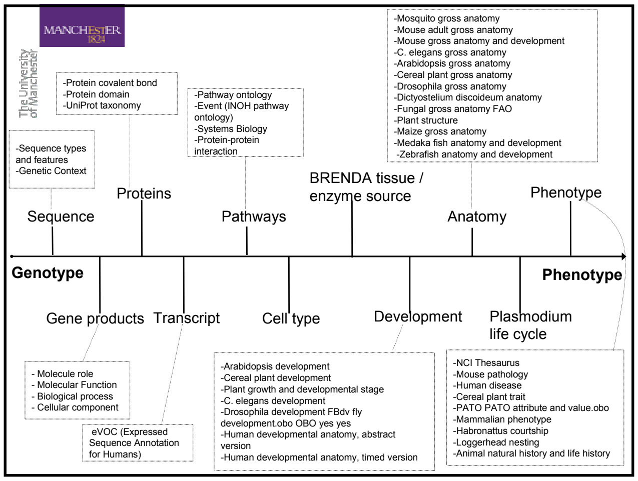
Talk Outline

- Introduction to OWL
- Representing biological knowledge in OWL
- A case study - the phosphatase example
- Ontological design patterns for the biologist
- Limitations posed by OWL
- Summary



Talk Aims

- To provide an insight into how OWL's model matches some of the requirements of the domain of biology
- To illustrate the design patterns that can be used to overcome some of the limitations of OWL
- To give a flavour of some of the 'hard' problems - the challenges posed by biology



Genotype

Phenotype

Sequence

-Sequence types and features
-Genetic Context

Proteins

-Protein covalent bond
-Protein domain
-UniProt taxonomy

Pathways

-Pathway ontology
-Event (INOH pathway ontology)
-Systems Biology
-Protein-protein interaction

BRENDA tissue / enzyme source

Anatomy

-Mosquito gross anatomy
-Mouse adult gross anatomy
-Mouse gross anatomy and development
-C. elegans gross anatomy
-Arabidopsis gross anatomy
-Cereal plant gross anatomy
-Drosophila gross anatomy
-Dictyostelium discoideum anatomy
-Fungal gross anatomy FAO
-Plant structure
-Maize gross anatomy
-Medaka fish anatomy and development
-Zebrafish anatomy and development

Phenotype

Gene products

- Molecule role
- Molecular Function
- Biological process
- Cellular component

Transcript

eVOC (Expressed Sequence Annotation for Humans)

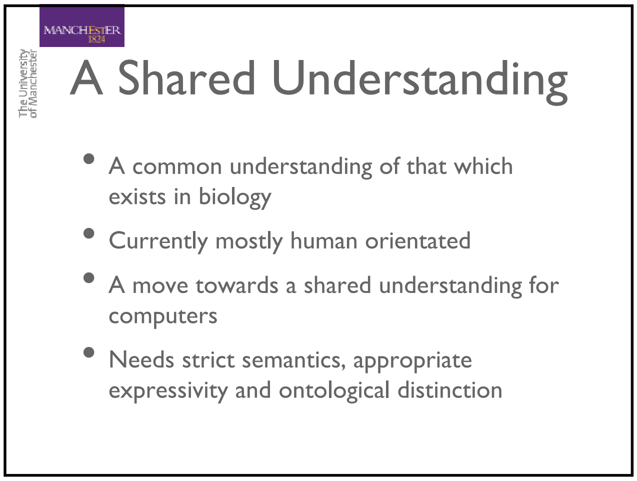
Cell type

Development

-Arabidopsis development
-Cereal plant development
-Plant growth and developmental stage
-C. elegans development
-Drosophila development FBdv fly development.obo OBO yes yes
-Human developmental anatomy, abstract version
-Human developmental anatomy, timed version

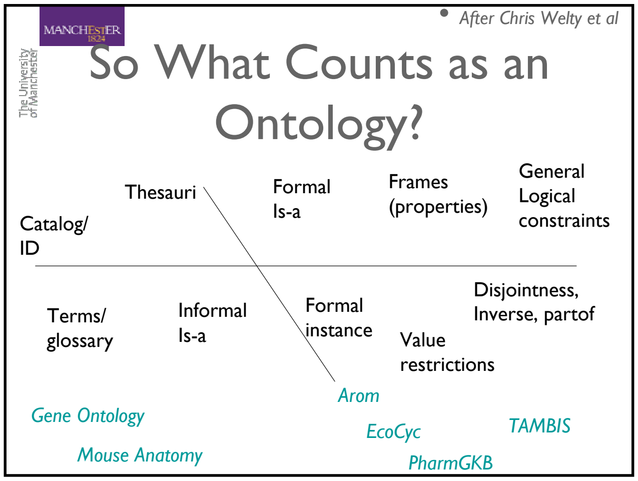
Plasmodium life cycle

-NCI Thesaurus
-Mouse pathology
-Human disease
-Cereal plant trait
-PATO PATO attribute and value.obo
-Mammalian phenotype
-Habronattus courtship
-Loggerhead nesting
-Animal natural history and life history



A Shared Understanding

- A common understanding of that which exists in biology
- Currently mostly human orientated
- A move towards a shared understanding for computers
- Needs strict semantics, appropriate expressivity and ontological distinction



- *After Chris Welty et al*

So What Counts as an Ontology?

Catalog/ ID

Thesauri

Formal Is-a

Frames (properties)

General Logical constraints

Terms/ glossary

Informal Is-a

Formal instance

Value restrictions

Disjointness, Inverse, partof

Gene Ontology

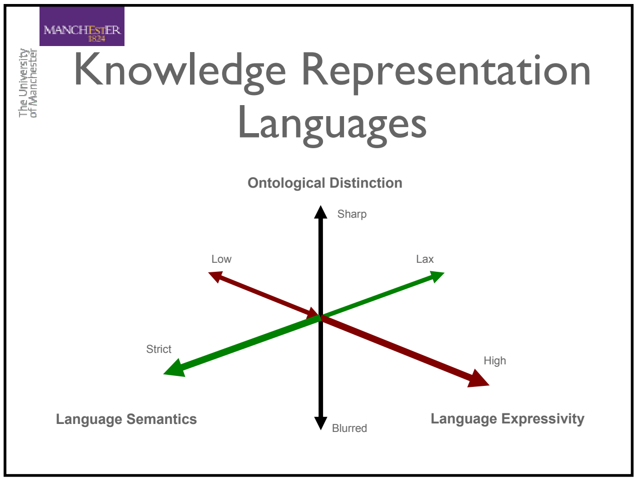
Mouse Anatomy

Arom

EcoCyc

PharmGKB

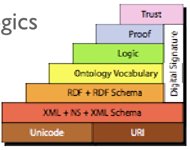
TAMBIS



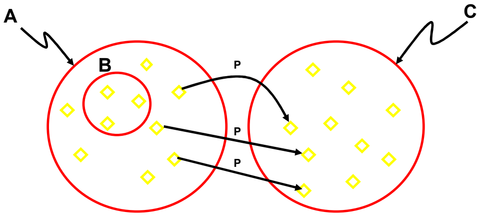
Knowledge Representation Languages

OWL

- Ontologies will form the back bone of the semantic web
- OWL is the latest standard in ontology languages from the W3C
- Layered on top of RDF and RDF Schema
- Underpinned by Description Logics



OWL in One Slide



Description Logics

- A decidable fragment of First Order Logic
- Well defined & strict semantics
- Possible to use machine reasoning:

–Make implicit knowledge explicit

–Aid the construction of an ontology

- Reasoning services provided by DL reasoners include:

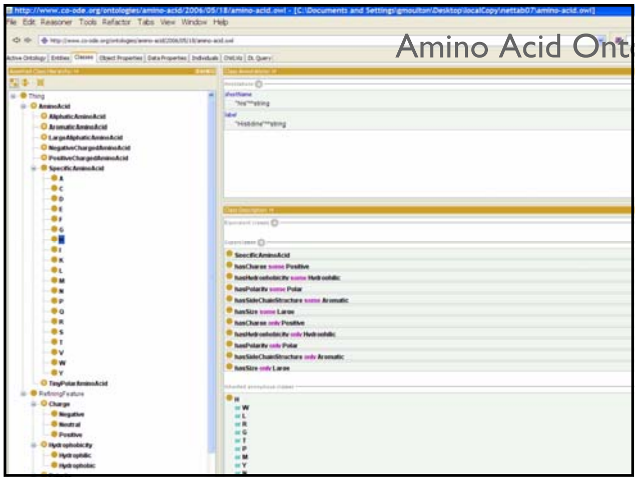
–Subsumption

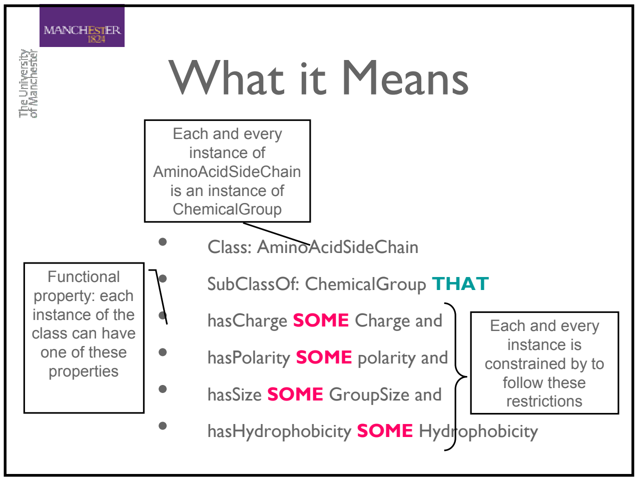
–Equivalence

–Consistency

–Instantiation

Amino Acid Ont





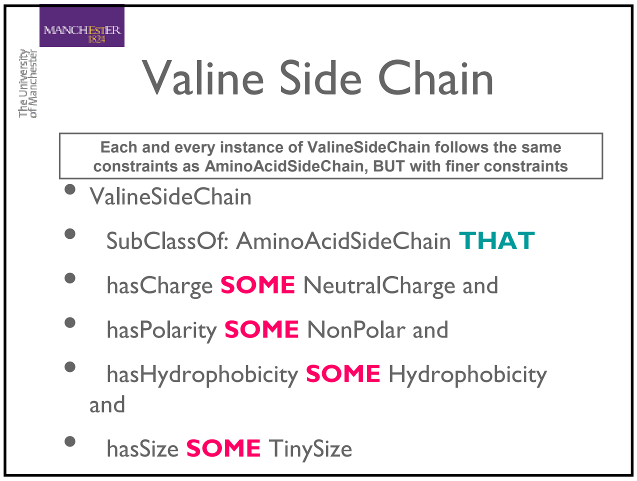
What it Means

Each and every instance of AminoAcidSideChain is an instance of ChemicalGroup

Functional property: each instance of the class can have one of these properties

Each and every instance is constrained by to follow these restrictions

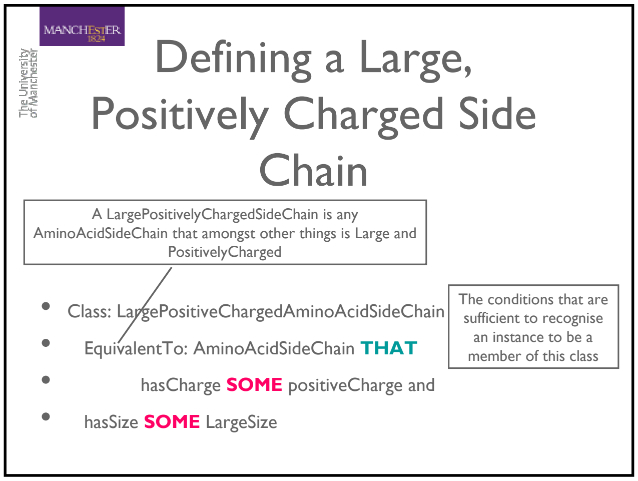
- Class: AminoAcidSideChain
- SubClassOf: ChemicalGroup **THAT**
- hasCharge **SOME** Charge and
- hasPolarity **SOME** polarity and
- hasSize **SOME** GroupSize and
- hasHydrophobicity **SOME** Hydrophobicity



Valine Side Chain

Each and every instance of ValineSideChain follows the same constraints as AminoAcidSideChain, BUT with finer constraints

- ValineSideChain
- SubClassOf: AminoAcidSideChain **THAT**
- hasCharge **SOME** NeutralCharge and
- hasPolarity **SOME** NonPolar and
- hasHydrophobicity **SOME** Hydrophobicity and
- hasSize **SOME** TinySize

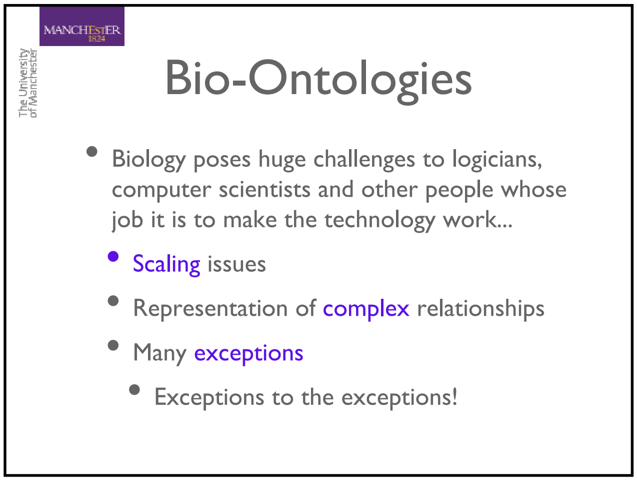


Defining a Large, Positively Charged Side Chain

A LargePositivelyChargedSideChain is any AminoAcidSideChain that amongst other things is Large and PositivelyCharged

The conditions that are sufficient to recognise an instance to be a member of this class

- Class: LargePositiveChargedAminoAcidSideChain
- EquivalentTo: AminoAcidSideChain **THAT**
- hasCharge **SOME** positiveCharge and
- hasSize **SOME** LargeSize



Bio-Ontologies

- Biology poses huge challenges to logicians, computer scientists and other people whose job it is to make the technology work…
  - Scaling issues
  - Representation of complex relationships
  - Many exceptions
    - Exceptions to the exceptions!

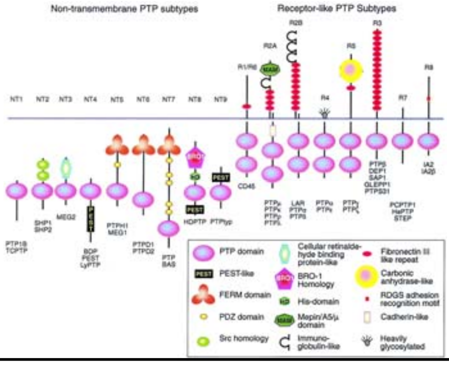
A Case Study

- A peek at how OWL can successfully be used to model biological knowledge
- Motivation: Use OWL to automate the classification of proteins from new genomic sequences

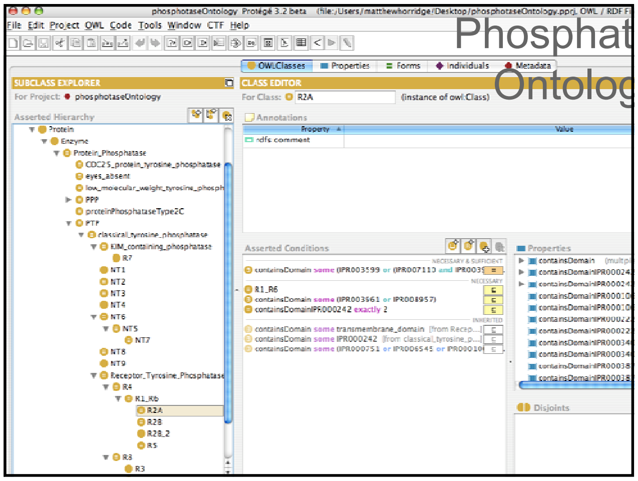
Protein Classification

- Bioinformaticians use tools to identify functional domains (e.g., InterProScan)
- Tools simply show the presence of domains - they do not classify proteins
- Experts classify proteins according to domain arrangements - the presence and number of each domain is important

Phosphatase Functional Domains



Phosphat Ontolog


phosphotaseOntology Protégé 3.2 beta (file:/Users/matthewhorridge/Desktop/phosphotaseOntology.pprj, OWL / RDF Fi
File Edit Project OWL Code Tools Window CTF Help
OWLClasses
Properties
Forms
Individuals
Metadata
SUBCLASS EXPLORER
CLASS EDITOR
Asserted Hierarchy
Annotations
Property
Value
Asserted Conditions
Properties
Disjoints

For Project: phosphotaseOntology

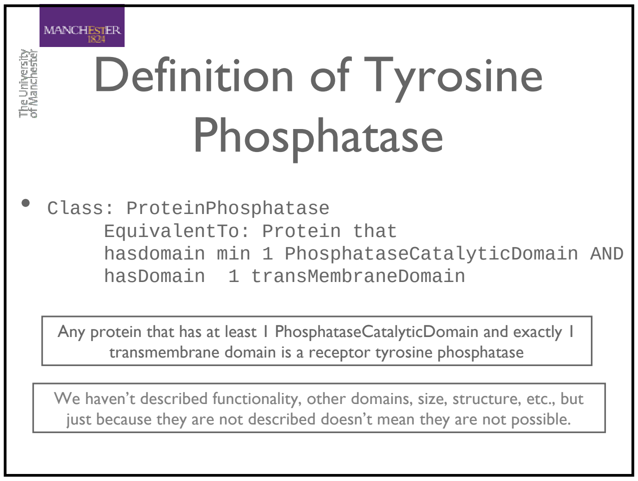
For Class: R2A (instance of owl:Class)

```
Protein
  Enzyme
    Protein_Phosphatase
      CDC25_protein_tyrosine_phosphatase
      eyes_absent
      low_molecular_weight_tyrosine_phosph
      PPP
      proteinPhosphataseType2C
      PTP
        classical_tyrosine_phosphatase
          KIM_containing_phosphatase
            R7
          NT1
          NT2
          NT3
          NT4
          NT6
            NT5
              NT7
          NT8
          NT9
          Receptor_Tyrosine_Phosphatase
            R4
              R1_R6
                R2A
                R2B
                R2B_2
                R5
            R3
              R3
```

rdfs:comment

```
NECESSARY & SUFFICIENT
containsDomain some (IPR003599 or (IPR007110 and IPR0039
NECESSARY
R1_R6
containsDomain some (IPR003961 or IPR008957)
containsDomainIPR000242 exactly 2
INHERITED
containsDomain some transmembrane_domain [from Recep...]
containsDomain some IPR000242 [from classical_tyrosine_p...]
containsDomain some (IPR000751 or IPR006545 or IPR00010
```

```
containsDomain (multipl
containsDomainIPR00024
containsDomainIPR00024
containsDomainIPR00010
containsDomainIPR00010
containsDomainIPR00022
containsDomainIPR00022
containsDomainIPR00034
containsDomainIPR00034
containsDomainIPR00038
containsDomainIPR00038
```

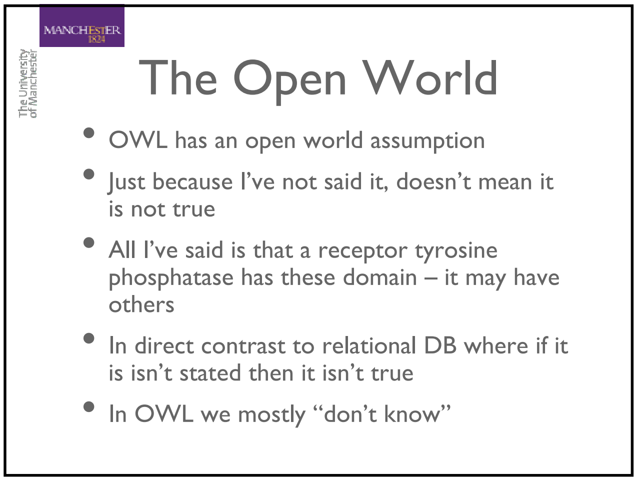


Definition of Tyrosine Phosphatase

```
• Class: ProteinPhosphatase
       EquivalentTo: Protein that
       hasdomain min 1 PhosphataseCatalyticDomain AND
       hasDomain  1 transMembraneDomain
```

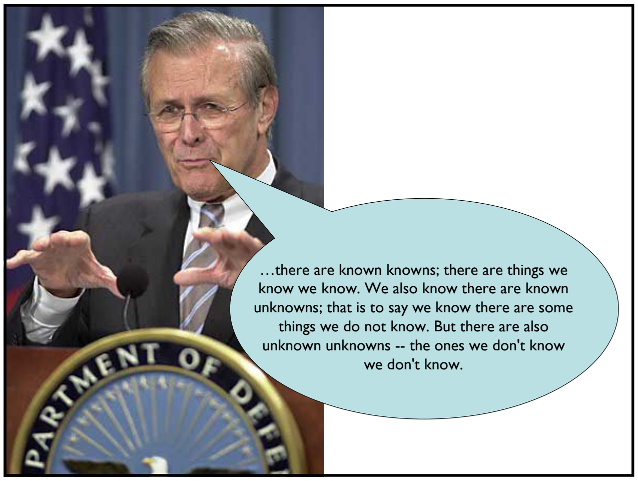
Any protein that has at least 1 PhosphataseCatalyticDomain and exactly 1 transmembrane domain is a receptor tyrosine phosphatase

We haven't described functionality, other domains, size, structure, etc., but just because they are not described doesn't mean they are not possible.

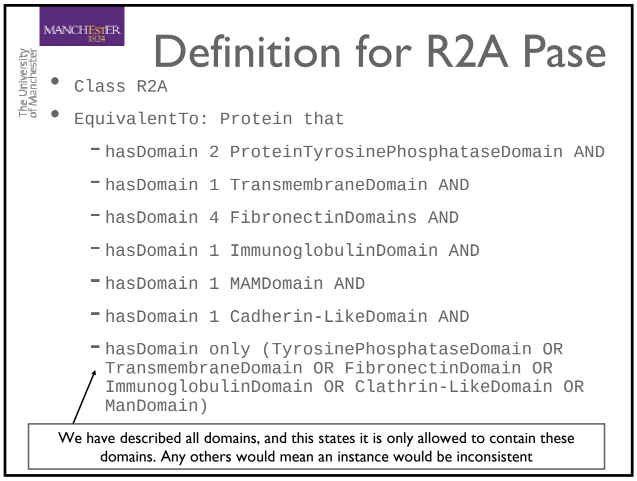


The Open World

- OWL has an open world assumption
- Just because I've not said it, doesn't mean it is not true
- All I've said is that a receptor tyrosine phosphatase has these domain – it may have others
- In direct contrast to relational DB where if it is isn't stated then it isn't true
- In OWL we mostly "don't know"



…there are known knowns; there are things we know we know. We also know there are known unknowns; that is to say we know there are some things we do not know. But there are also unknown unknowns -- the ones we don't know we don't know.



Definition for R2A Pase

```
• Class R2A
• EquivalentTo: Protein that
   - hasDomain 2 ProteinTyrosinePhosphataseDomain AND
   - hasDomain 1 TransmembraneDomain AND
   - hasDomain 4 FibronectinDomains AND
   - hasDomain 1 ImmunoglobulinDomain AND
   - hasDomain 1 MAMDomain AND
   - hasDomain 1 Cadherin-LikeDomain AND
   - hasDomain only (TyrosinePhosphataseDomain OR
     TransmembraneDomain OR FibronectinDomain OR
     ImmunoglobulinDomain OR Clathrin-LikeDomain OR
     ManDomain)
```

We have described all domains, and this states it is only allowed to contain these domains. Any others would mean an instance would be inconsistent

Qualified Cardinality Constraints

- Restrictions are often just existential
- At least one of the successor
- Can specify how many instances are involved by qualifying the cardinality
- hasDomain 2 FibronectinDomain
- Min-2, max-4, etc.
- OWL 1.0 didn't have QCR, though the reasoners could use it

Description of an Instance of a Protein

- Instance: P21592

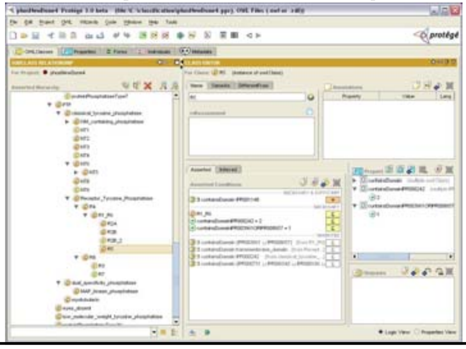
```
TypeOf: Protein That
Fact: hasDomain 2
ProteinTyrosinePhosphataseDomain and
Fact: hasdomain 1 TransmembraneDomain
and
Fact: hasdomain 4 FibronectinDomains
and
Fact: hasDomain 1
ImmunoglobulinDomain and
Fact: hasdomain 1 MAMDomain and
Fact: hasdomain 1 Cadherin-LikeDomain
```

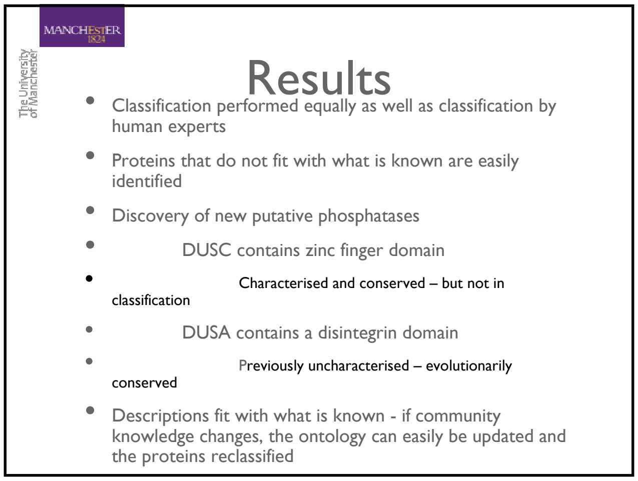
Tyrosine Phosphatase
(containsDomain some TransmembraneDomain) and
(containsDomain at least 1 ProteinTyrosinePhosphataseDomain)

R2A Instance: P21592
TypeOf: Protein That
Fact: hasDomain 2
ProteinTyrosinePhosphataseDomain and
Fact: hasdomain 1 TransmembraneDomain and
Fact: hasdomain 4 FibronectinDomains and
Fact: hasDomain 1 ImmunoglobulinDomain and
Fact: hasdomain 1 MAMDomain and
Fact: hasdomain 1 Cadherin-LikeDomain

R2A Phosphatase
(containsDomain some MAMDomain) and
(containsDomain some ProteinTyrosineCatalyticDomain or ImmunoglobulinDomain) and
(containsDomain some FibronectinDomain or FibronectinTypeIIIFoldDomain) and
(containsDomain exactly 2 ProteinTyrosinePhosphataseDomain)

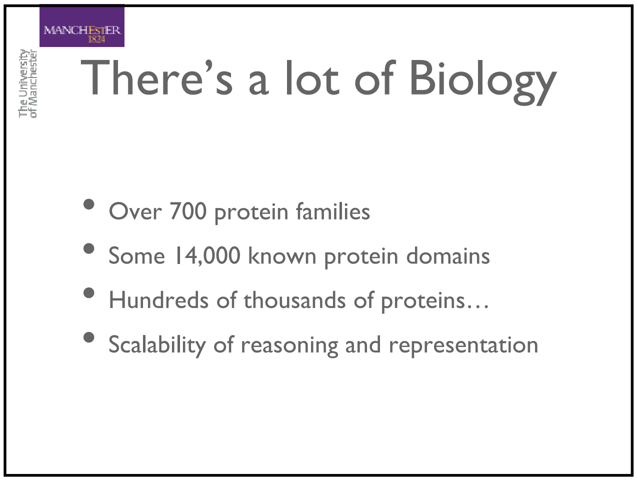
Classification of Protein Tyrosine Phosphatases





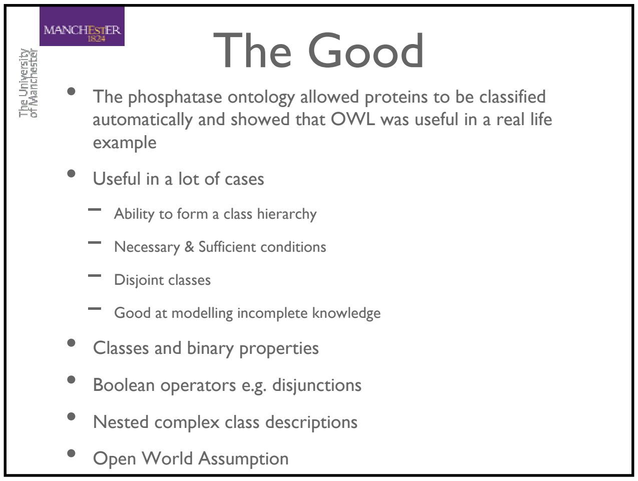
Results

- Classification performed equally as well as classification by human experts
- Proteins that do not fit with what is known are easily identified
- Discovery of new putative phosphatases
- DUSC contains zinc finger domain
- Characterised and conserved – but not in classification
- DUSA contains a disintegrin domain
- Previously uncharacterised – evolutionarily conserved
- Descriptions fit with what is known - if community knowledge changes, the ontology can easily be updated and the proteins reclassified



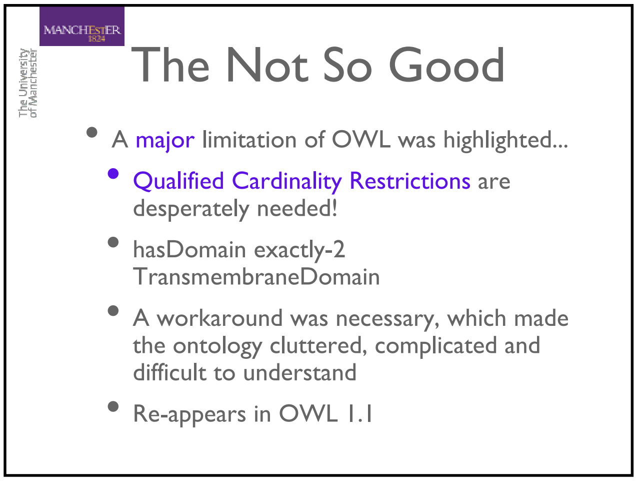
There’s a lot of Biology

- Over 700 protein families
- Some 14,000 known protein domains
- Hundreds of thousands of proteins…
- Scalability of reasoning and representation



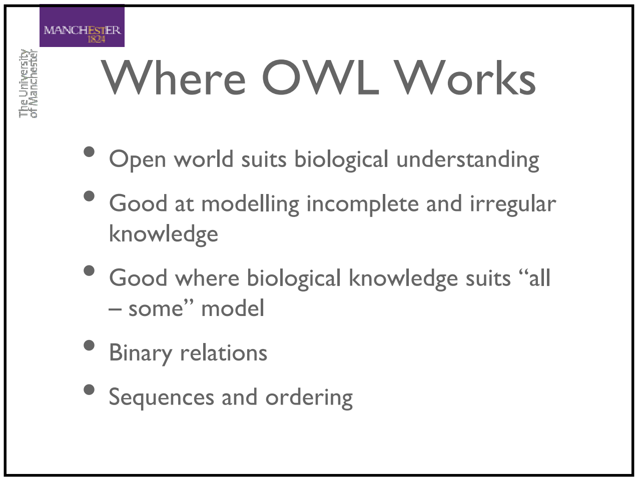
The Good

- The phosphatase ontology allowed proteins to be classified automatically and showed that OWL was useful in a real life example
- Useful in a lot of cases
  - Ability to form a class hierarchy
  - Necessary & Sufficient conditions
  - Disjoint classes
  - Good at modelling incomplete knowledge
- Classes and binary properties
- Boolean operators e.g. disjunctions
- Nested complex class descriptions
- Open World Assumption



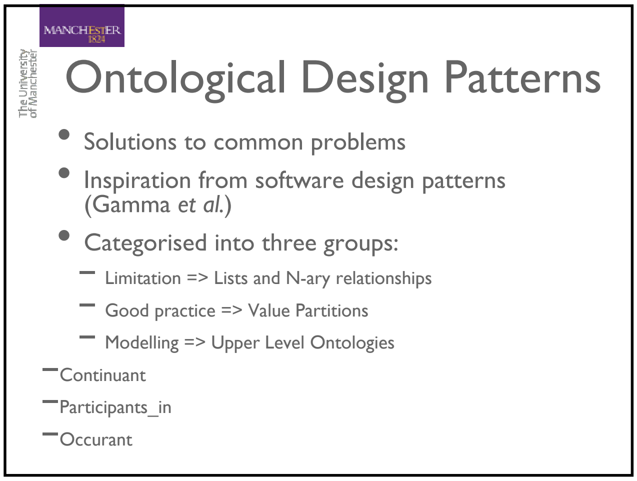
The Not So Good

- A major limitation of OWL was highlighted...
  - Qualified Cardinality Restrictions are desperately needed!
  - hasDomain exactly-2 TransmembraneDomain
  - A workaround was necessary, which made the ontology cluttered, complicated and difficult to understand
  - Re-appears in OWL 1.1



Where OWL Works

- Open world suits biological understanding
- Good at modelling incomplete and irregular knowledge
- Good where biological knowledge suits “all – some” model
- Binary relations
- Sequences and ordering



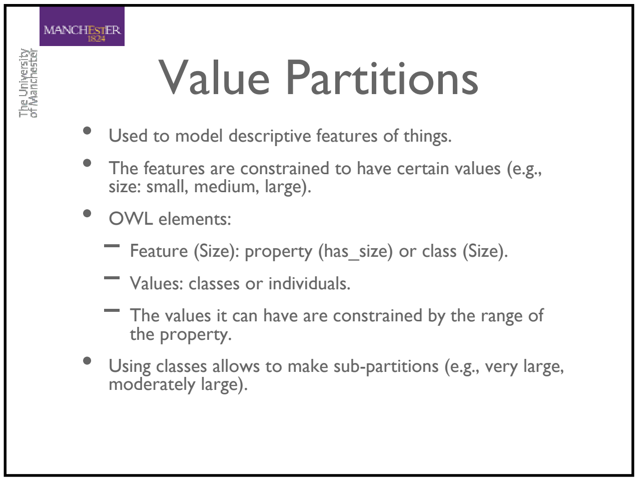
Ontological Design Patterns

- Solutions to common problems
- Inspiration from software design patterns (Gamma *et al.*)
- Categorised into three groups:
  - Limitation => Lists and N-ary relationships
  - Good practice => Value Partitions
  - Modelling => Upper Level Ontologies

– Continuant

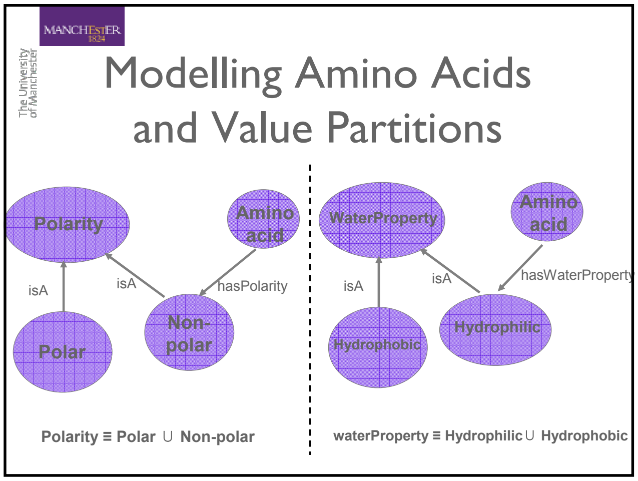
– Participants_in

– Occurant



Value Partitions

- Used to model descriptive features of things.
- The features are constrained to have certain values (e.g., size: small, medium, large).
- OWL elements:
  - Feature (Size): property (has_size) or class (Size).
  - Values: classes or individuals.
  - The values it can have are constrained by the range of the property.
- Using classes allows to make sub-partitions (e.g., very large, moderately large).



Modelling Amino Acids and Value Partitions

Polarity ≡ Polar ∪ Non-polar

waterProperty ≡ Hydrophilic ∪ Hydrophobic

Protégé and Value Partitions

- Value Partition

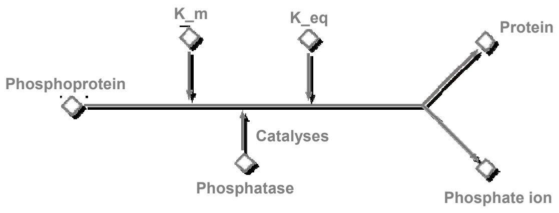
Design Patterns in Biology

- Representation of n-ary relations
- Representation of exceptions
- Representation of ordering using lists

N-ary Relations

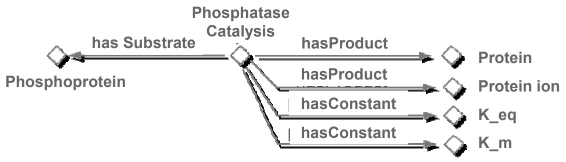
- OWL properties are interpreted as binary relations on individuals - i.e. sets of pairs of individuals
- We often need higher arity relations that link more than two individuals
- For example we would like to talk about the catalysis of phosphoproteins

N-ary Relations



N-ary Relations in OWL

- n-ary relations are simulated in OWL by turning the property into a class that represents the relation
- N-aryRelationships

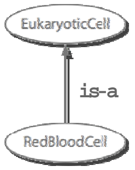


Exceptions

- We have already established the fact that OWL-DL talks about what is universally true of a class of individuals
- Classic example of all birds fly (except ostrich, ...)
- Biology is supposedly full of exceptions
- All eukaryotic cells have a nucleus

Exception Example

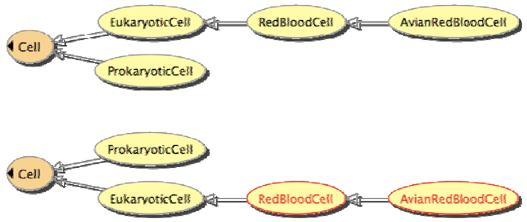
- All eukaryotic cell have one nucleus,
- Mammalian red blood cells don't have nucleus but they are eukaryotic cells
- Avian red cells do
- Some cells are polynucleate



hasNucleus min 1

hasNucleus min 0

RBC and Avian RBC Example

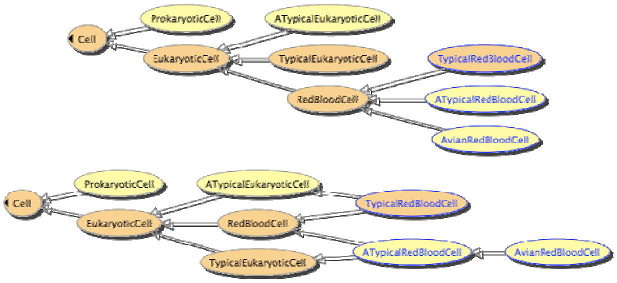


Exceptions Pattern

For any exception class X,

- Create two subclasses of X, one TypicalX, one representing AtypicalX
- Add a covering axiom to X to state that instances of X are either typical or atypical
- The conditions that make X typical are pushed down into TypicalX
- All other subclasses of X are left unchanged

Cell Example (Asserted/Inferred)

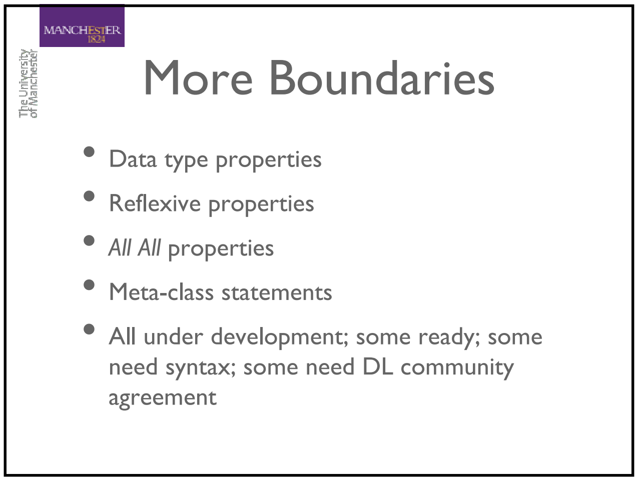


Exception Pattern

- The exception pattern allows us to compensate for the fact that OWL talks about what is universally true - conditions hold for all instances of a class
- The pattern is messy:
  - Requires auxiliary classes that clutter up the hierarchy
  - Unintuitive to domain experts like biologists

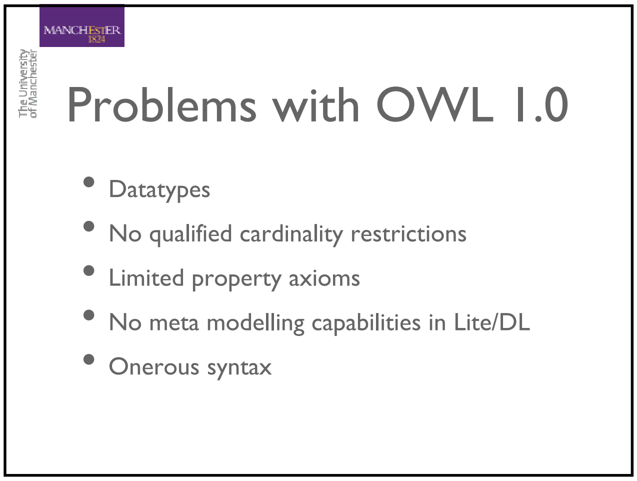
The Boundaries of OWL 1.0

- No qualified cardinality restrictions
- Defaults and exceptions
- Complex property restrictions
- Expressive data types
- Fuzziness, probability and similarity



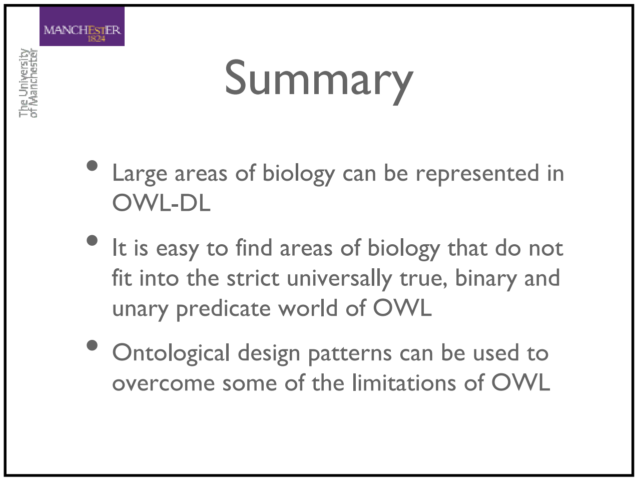
More Boundaries

- Data type properties
- Reflexive properties
- *All All* properties
- Meta-class statements
- All under development; some ready; some need syntax; some need DL community agreement



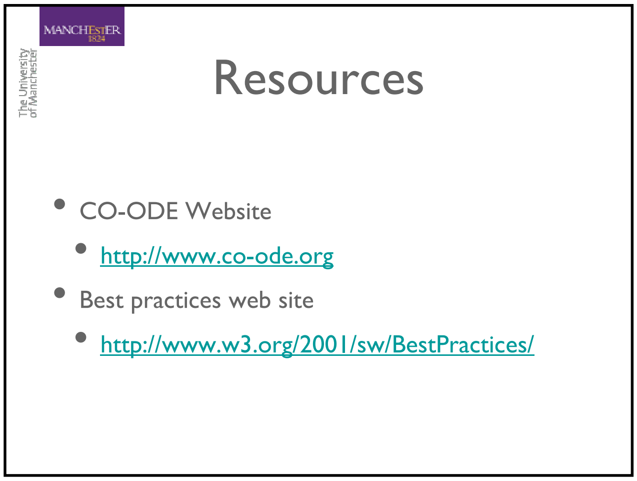
Problems with OWL 1.0

- Datatypes
- No qualified cardinality restrictions
- Limited property axioms
- No meta modelling capabilities in Lite/DL
- Onerous syntax



Summary

- Large areas of biology can be represented in OWL-DL
- It is easy to find areas of biology that do not fit into the strict universally true, binary and unary predicate world of OWL
- Ontological design patterns can be used to overcome some of the limitations of OWL



Resources

- CO-ODE Website
  - http://www.co-ode.org
- Best practices web site
  - http://www.w3.org/2001/sw/BestPractices/

OWL 1.1 Philosophy

- Simple extension of OWL-DL
- Maintain decidability of the language
- Focus on features for which useful reasoning techniques are known and which are likely to be implemented
- Theoretical worst-case complexity high (as in OWL-DL)
- Based on `SROIQ` description logic

Not Included

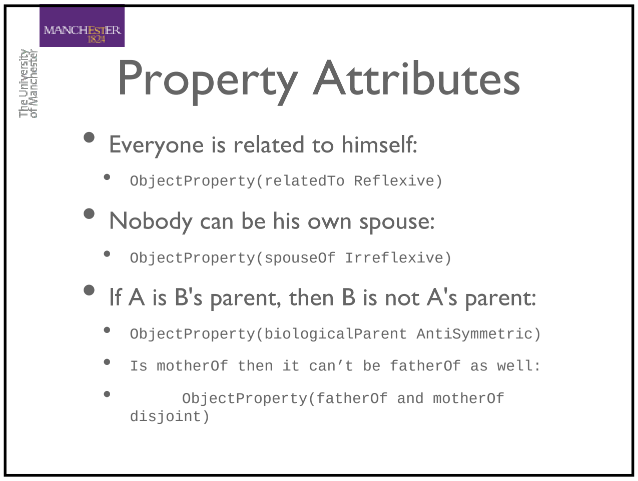
- Non-monotonic extensions
- Rules language
- Temporal and spatial constructs
- Probabilistic and fuzzy extensions
- Query languages/explanation

New OWL 1.1 Features

- Qualified cardinality restrictions
- Additional property types (reflexive, anti-symmetric)
- Disjoint properties
- Property chain inclusion axioms
- User-defined data-types and data-type predicates
- Limited form of meta-modelling
- Syntactic sugar

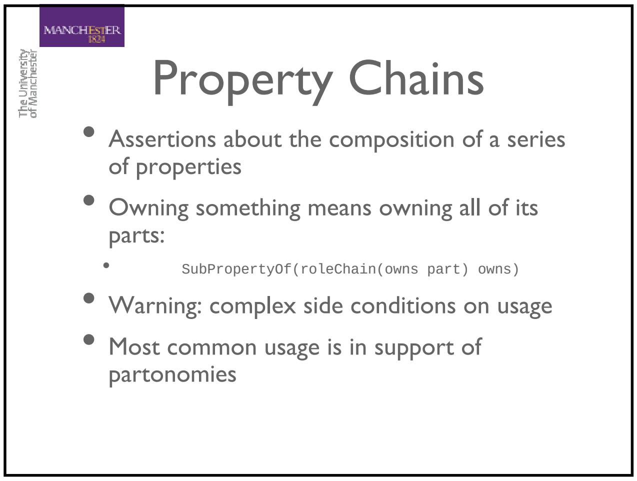
Qualified Number Restrictions

- The heart has four chambers: two atria and two ventricles
  - `Class(Heart partial restriction(hasChamber cardinality(4)))`
  - `Class(Heart partial restriction(hasChamber cardinality(2 atrium)))`
  - `Class(Heart partial restriction(hasChamber cardinality(2 ventricle)))`
- A medical oversight committee must have at least two medically-qualified members
  - `Class(MedicalOversightCommittee partial`
  - `        restriction(hasMember minCardinality(2 Doctor)))`
- A legal drug regimen must not contain more than one Central Nervous System depressant, although it may contain any number of drugs in total:
  - `Class(LegalDrugRegimen partial`
  - `        restriction(includesDrug maxCardinality(1 CNS-Depressant)))`



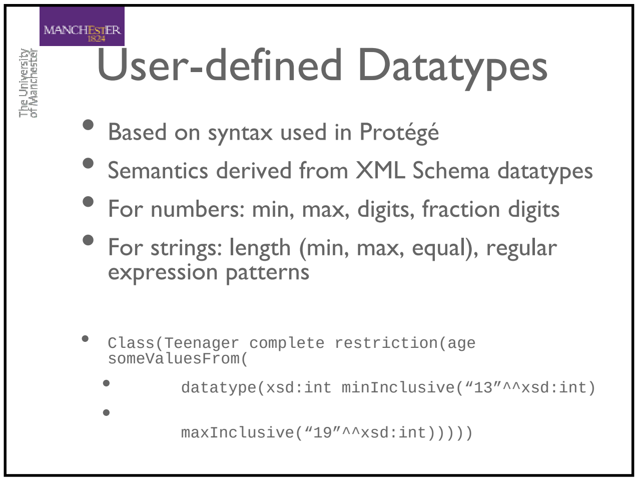
Property Attributes

- Everyone is related to himself:
  - `ObjectProperty(relatedTo Reflexive)`
- Nobody can be his own spouse:
  - `ObjectProperty(spouseOf Irreflexive)`
- If A is B's parent, then B is not A's parent:
  - `ObjectProperty(biologicalParent AntiSymmetric)`
  - `Is motherOf then it can’t be fatherOf as well:`
  - `ObjectProperty(fatherOf and motherOf disjoint)`



Property Chains

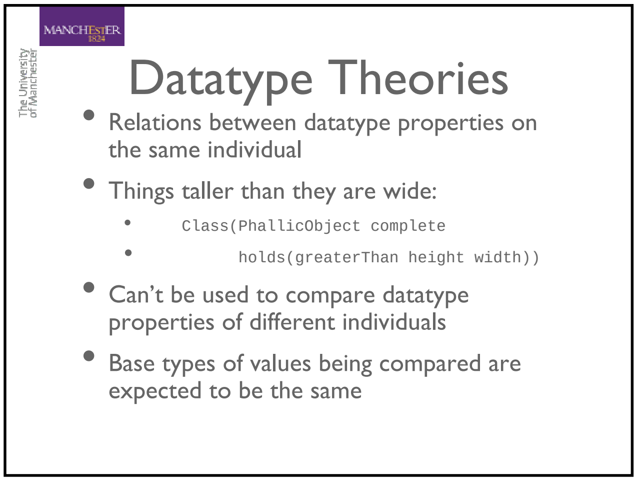
- Assertions about the composition of a series of properties
- Owning something means owning all of its parts:
  - `SubPropertyOf(roleChain(owns part) owns)`
- Warning: complex side conditions on usage
- Most common usage is in support of partonomies



User-defined Datatypes

- Based on syntax used in Protégé
- Semantics derived from XML Schema datatypes
- For numbers: min, max, digits, fraction digits
- For strings: length (min, max, equal), regular expression patterns

- `Class(Teenager complete restriction(age someValuesFrom(`
  - `datatype(xsd:int minInclusive(“13”^^xsd:int)`
  - `maxInclusive(“19”^^xsd:int)))))`



Datatype Theories

- Relations between datatype properties on the same individual
- Things taller than they are wide:
  - `Class(PhallicObject complete`
  - `holds(greaterThan height width))`
- Can’t be used to compare datatype properties of different individuals
- Base types of values being compared are expected to be the same

Punning

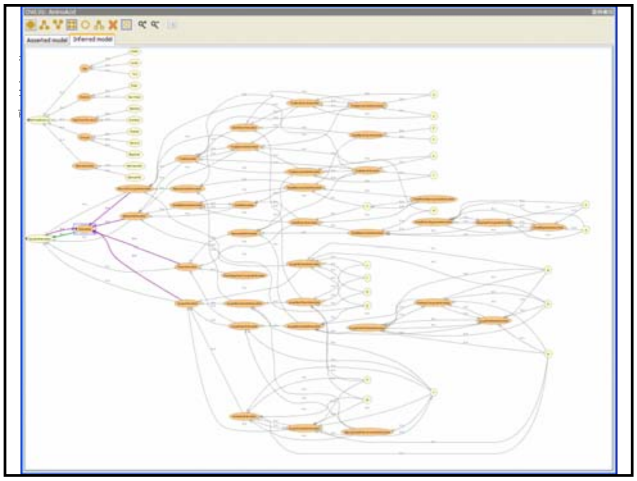
- In OWL-DL, a name refers to either a class, a property, or an individual
- In OWL 1.1, the same name can be used for each of these independently; there is no connection between the three namespaces
  - `Class(Person)`
  - `Individual(Person)`
  - `Individual(John Person)`
  - `SameIndividualAs(Person Rock)`
- This does *not* imply
  - `Individual(John Rock)`
- Incompatible with RDF

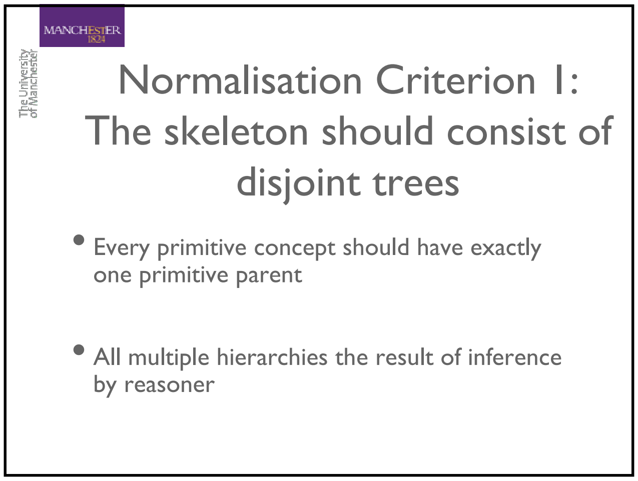
Meta-modelling

- Punning provides a convenient way to attach properties to class names
  - `Individual(John)`
  - `Class(Person)`
  - `ObjectProperty(createdBy range(Person))`
  - `Individual(Person restriction(createdBy value(John)))`
- rdfs:label and rdfs:comment are data-valued properties in OWL 1.1

Rationale for Normalisation

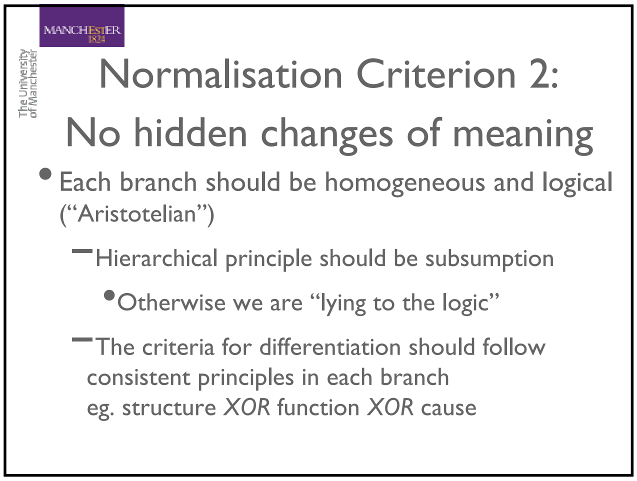
- Maintenance
  - Each change in exactly one place
  - No "Side effects"
- Modularisation
  - Each primitive must belong to exactly one module
    - *If a primitive belongs to two modules, they are not modular.*
    - *If a primitive belongs to two modules, it probably conflates two notions*
  - concentrate on the *"primitive skeleton"* of the domain ontology
- Parsimony
  - Requires fewer axioms





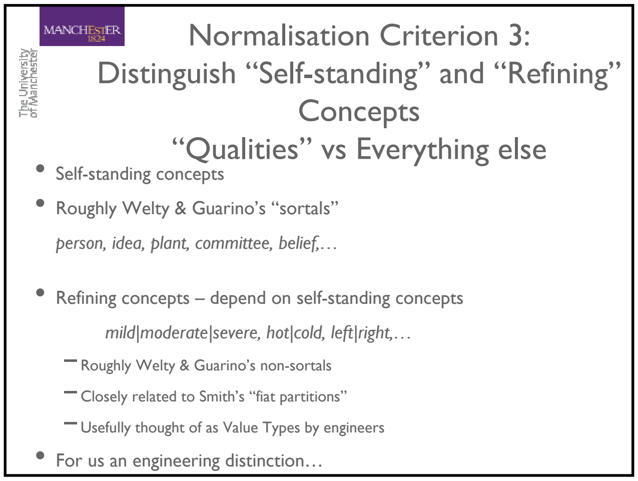
Normalisation Criterion 1: The skeleton should consist of disjoint trees

- Every primitive concept should have exactly one primitive parent
- All multiple hierarchies the result of inference by reasoner



Normalisation Criterion 2: No hidden changes of meaning

- Each branch should be homogeneous and logical (“Aristotelian”)
  - Hierarchical principle should be subsumption
    - Otherwise we are “lying to the logic”
  - The criteria for differentiation should follow consistent principles in each branch eg. structure *XOR* function *XOR* cause



Normalisation Criterion 3: Distinguish “Self-standing” and “Refining” Concepts “Qualities” vs Everything else

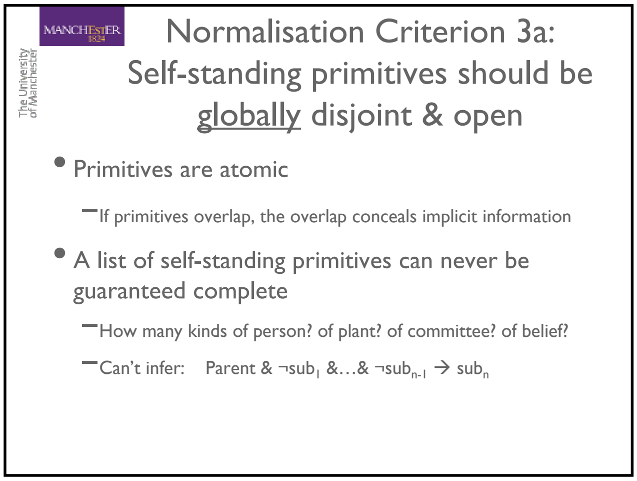
- Self-standing concepts
- Roughly Welty & Guarino’s “sortals”

  person, idea, plant, committee, belief,…

- Refining concepts – depend on self-standing concepts

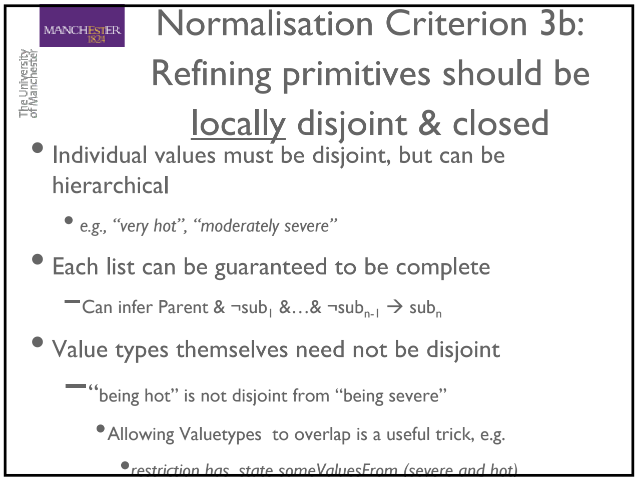
  mild|moderate|severe, hot|cold, left|right,…

  - Roughly Welty & Guarino’s non-sortals
  - Closely related to Smith’s “fiat partitions”
  - Usefully thought of as Value Types by engineers
- For us an engineering distinction…



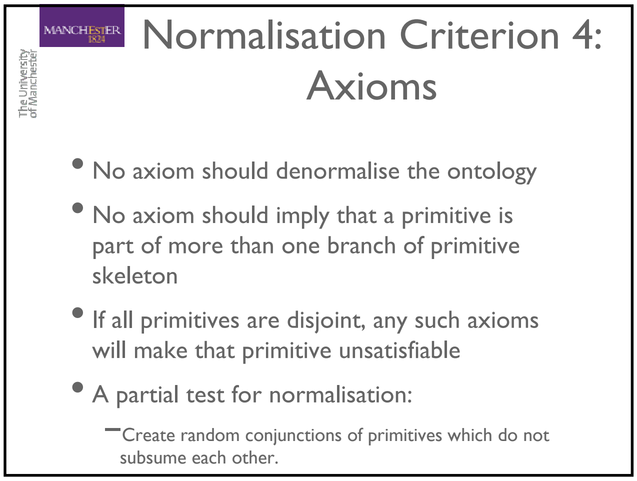
Normalisation Criterion 3a: Self-standing primitives should be <u>globally</u> disjoint & open

- Primitives are atomic
  - If primitives overlap, the overlap conceals implicit information
- A list of self-standing primitives can never be guaranteed complete
  - How many kinds of person? of plant? of committee? of belief?
  - Can’t infer: Parent & ¬sub_1 &…& ¬sub_{n-1} → sub_n



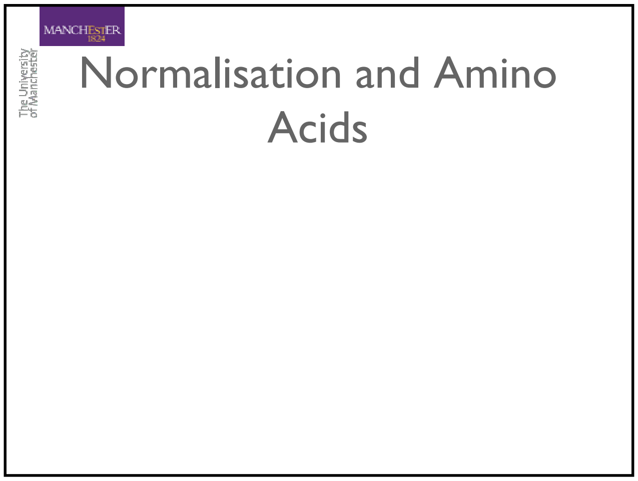
Normalisation Criterion 3b: Refining primitives should be locally disjoint & closed

- Individual values must be disjoint, but can be hierarchical
  - *e.g., "very hot", "moderately severe"*
- Each list can be guaranteed to be complete
  - Can infer Parent & $\neg sub_1$ &…& $\neg sub_{n-1}$ → sub_n
- Value types themselves need not be disjoint
  - "being hot" is not disjoint from "being severe"
    - Allowing Valuetypes to overlap is a useful trick, e.g.
      - *restriction has_state someValuesFrom (severe and hot)*



Normalisation Criterion 4: Axioms

- No axiom should denormalise the ontology
- No axiom should imply that a primitive is part of more than one branch of primitive skeleton
- If all primitives are disjoint, any such axioms will make that primitive unsatisfiable
- A partial test for normalisation:
  - Create random conjunctions of primitives which do not subsume each other.



Normalisation and Amino Acids