OWL Pizzas:
Practical Experience of Teaching OWL-DL:
Common Errors & Common Patterns

Alan Rector[1], Nick Drummond[1], Matthew Horridge[1], Jeremy Rogers[1], Holger Knublauch[2], Robert Stevens[1], Hai Wang[1], Chris Wroe[1]

[1]Information Management Group / Bio Health Informatics Forum
Department of Computer Science, University of Manchester

[2]Stanford Medical Informatics, Stanford University

rector@cs.man.ac.uk
co-ode-admin@cs.man.ac.uk

www.co-ode.org
protege.stanford.org

Why do so few people use OWL and DLs?
Why so little use of classifiers?
Is part of the answer that…

- **OWL/DLs run counter to common intuitions from**
  - **Databases, UML, query languages (including RDQL)**
  - **Logic programming & rule systems, e.g. JESS, PAL**
  - **Frame systems – more difference than at first appears**
  - **Object oriented programming**
- **Can Tools can help?**
  - **Can we use tutorials and training to gather requirement?**
    - **All examples here have occurred repeatedly in practice in tutorials or in live ontology construction – often by experts in other formalisms**
      - **Part of the requirements gathering for the Protégé-OWL interface**

OWL Pizzas Tutorial

- **Designed to address common errors**
  - **We have seen lots of experienced people make the same simple mistakes**
- **Why Pizzas?**
  - **Naturally combinatorial**
  - **No serious ontological issues**
  - **Familiar and fun (at least to western audiences)**
  - **Easy to illustrate most problems**
- **Extended version**
  - **See 120 pg 'textbook' version on http://www.co-ode.org**

Issues and common errors

- **Open world reasoning**
  - **Domain and range constraints as axioms**
  - **Trivial satisfiability of universal restrictions**
  - **Subsumption ("is kind of") as necessary implication**
- **Unfamiliar constructs – confusing notation/terminology**
  - **Confusion of universal (*allValuesFrom*) rather than existential restrictions (*someValuesFrom*)**
  - **Need for explicit disjointness axioms**
- **Errors in understanding common logical constructs**
  - **Confusing 'and' and 'or'**
  - **Defined vs primitive classes & conversion between them**
  - **Use of subclass axioms as rules**
- **Understanding the effect of classification**
  - **What to do when it all turns red – debugging**
  - **Explaining classification**

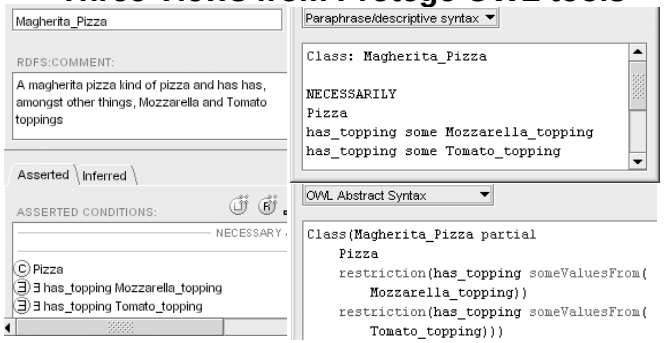
Open World Reasoning "Vegetarian Pizzas"

The menu says that:

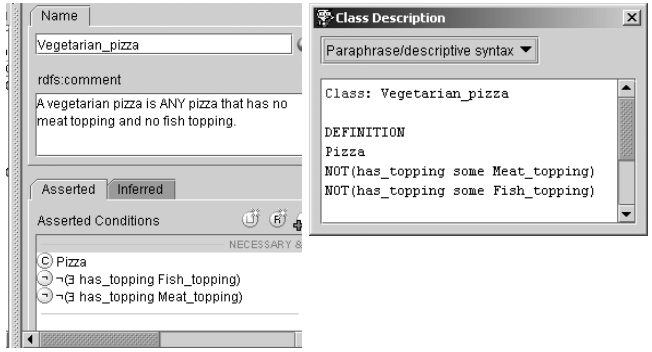
- **"Margherita pizzas have tomato and mozzarella toppings"**
- **"Vegetarian pizzas have no meat or fish toppings"**

What's it mean?

Three Views from Protégé OWL tools

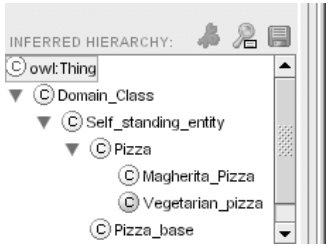


Vegetarian Pizza



Is a Margherita Pizza a Vegetarian Pizza?

- **Not according to classifier**



- **And not according to the full paraphrases formulated carefully**

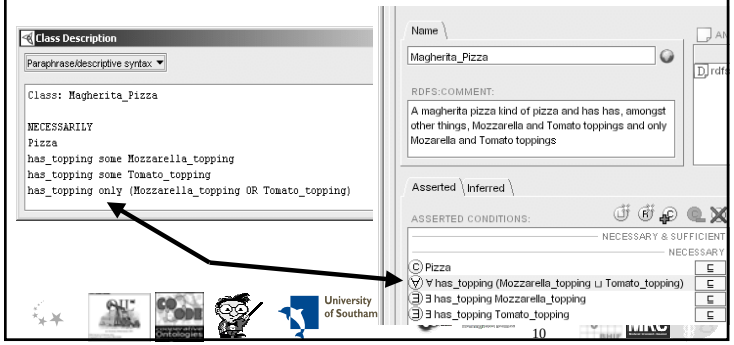
Open World Reasoning
Vegetarian & Margherita Pizzas

- **"A vegetarian pizza is *any* pizza that, *amongst other things,***
  does not* have *any* meat topping *and
  ***does not* have *any* fish topping"**

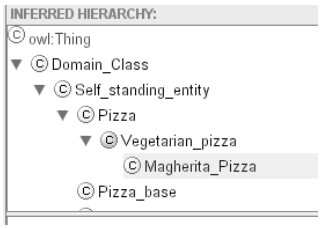
- **"A margherita pizza is *a* pizza and, *amongst other things,***
  has *some* tomato topping *and*
  has *some* mozarella topping"

Add "Closure Axiom"

- **"A Margherita pizza has tomato and cheese toppings and only tomato and cheese toppings"**
  - **i.e. "A Margherita pizza has tomato and cheese toppings and only toppings that are tomato *or* cheese"**
    - **Tedious to create by hand, so provide automatic generation in tool**



Now Classifies as Intended



- **Provided:**
  Toppings mutually disjoint

Domain & Range Constraints

- **Actually axioms**
  - ***Property P range(RangeClass*)**
    means
    - **owl:Thing**
      restriction(P *allValuesFrom* RangeClass)
  - ***Property P domain(DomainClass)***
    means
    - **owl:Thing**
      restriction(inverse(P) *allValuesFrom* DomainClass)

Non-Obvious Consequences

- **Range constraint violations – *unsatisfiable or ignored***
  - **If filler and RangeClass are disjoint: *unsatisfiable***
  - ***Otherwise nothing happens!***
- **Domain constraint violations – *unsatisfiable or coerced***
  - **If subject and DomainClass are disjoint: *unsatisfiable***
  - ***Otherwise, subject reclassified (coerced) to kind of DomainClass!***
- ***Furthermore cannot be fully checked before classification***
  - ***although tools can issue warnings.***

Example of Coercion by Domain violation

- **has_topping: *domain*(Pizza) *range*(Pizza_topping)**

  class Ice_cream_cone
  has_topping *some* Ice_cream

- **If Ice_cream_cone and Pizza are *not* disjoint:**
  - **Ice_cream_cone is classified as a kind of Pizza**
    …but: Ice_cream is *not* classified as a kind of Pizza_topping
    - **Have shown that:**
      ***all* Ice_cream_cones are a kinds of Pizzas,**
      but only that:
      ***some* Ice_cream is a kind of Pizza_topping**
      - » ***Only domain constraints can cause reclassification***
        … by now most people are very confused - need lots of examples & back to basics

Subsumption means necessary implication

- ***"B is a kind of A"***
  means
  "All Bs are As"
  - ***"Ice_cream_cone is a kind of Pizza"***
    means
    "All ice_cream_cones are pizzas"
    - **From "Some Bs are As" we can deduce very little of interest in DL terms**
      - » **"some ice_creams are pizza_toppings" says nothing about "all ice creams"**

Trivial Satisfiability: More unintuitive results

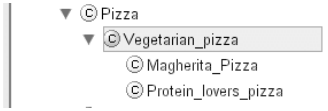
- **An existential (someValuesFrom) restriction with an empty filler makes no sense:**
  - **is unsatisfiable if its filler is unsatisfiable**
- **A Universal (allValuesFrom) restriction with an unsatisfiable filler is trivially satisfiable**
  - **provided there is no way to infer a existence of a filler**
    - **Leads to errors being missed and then appearing later**

Examples of Trivial Satisfaction

- **Unsatisfiable filler:**
  disjoint(Meat_topping Fish_topping)
  class(Protein_lovers_pizza complete
  has_topping *allValuesfrom* (Meat_topping *and* Fish_topping))
  - **i.e. intersectionOf(Meat_topping, Fish_topping)**
  - **i.e. *only something that is both* (Meat_topping *and* fish_topping)**
- **Range constraint violation:**
  disjoint(Ice_cream, Pizza_topping)
  class(Ice_cream_pizza
  has_topping *allValuesFrom* Ice_cream)
- **Both legal unless/until there is an axiom such as:**
  Pizza has_topping *someValuesFrom* Pizza_topping
  - **i.e. "All pizzas have at least one topping"**

Worse, Trivially Satisfied Restrictions Classify under Anything

- **Protein_lovers_pizza *is a kind of* Vegetarian_Pizza!**

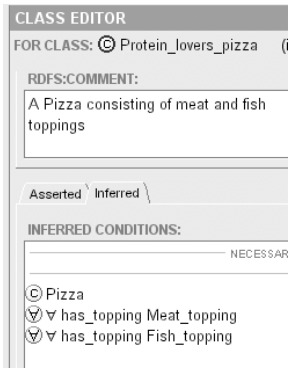


- **Until we add:**
  Pizza has_topping some Pizza_topping
  - **"All pizzas have some topping"**

"Only does not imply some!"

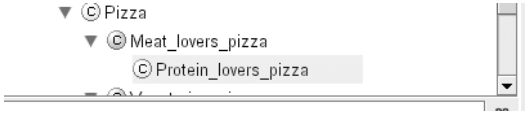
The trouble with confusing "some" with "only" *someValuesFrom* with *allValuesFrom*

- **It works for a while**
  - **The student defining Protein_lovers_pizza thought they were defining a pizza with meat toppings and fish toppings**
- **Errors only show up later when existentials are added elsewhere**



The trouble with confusing "some" with "only" *someValuesFrom* with *allValuesFrom*

- **Even classification seems to work at first**
  - **class(Meat_lovers_pizza complete**
    has_topping *only* Meat_topping)

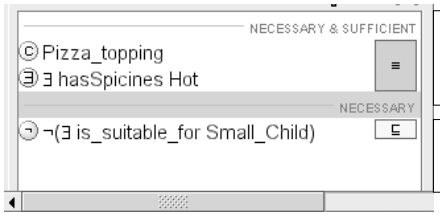


- **So people continue complacently**
  - **Until the unexpected happens, e.g.**
    - **It is also classified as a kind of vegetarian pizza**
    - **It is made unsatisfiable by an existential axiom someplace**

Defined vs Primitive Classes

- **In OWL the difference is a single keyword**
  - **“partial” vs “complete”**
- **In OilEd it was a single button**
  - **“subclass” vs “same class as” or “partial” vs “complete”**
- **Also…**
  Any necessary restrictions on defined classes must appear in separate subclassOf axioms
  - **Breaks the object oriented paradigm**
    - **Hides information about the class on a different pane**
  - **Makes migrating a primitive class to a defined class tedious**
    - **Unless all restrictions become part of the definition**
  - **Makes subclass axioms for implication hard to understand**

Protégé-OWL – Everything in one place



Necessary & Sufficient conditions:

“Definition”

Necessary conditions:

“Description”

- **Spicy_Pizza_topping**
  Necessary & Sufficient:
  Pizza_topping &
  has_spiciness *some* Hot
  Necessarily also
  ***Not* suitable_for *any* Small_child**

Defined classes

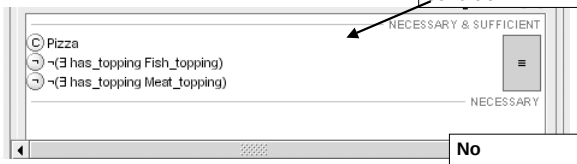
- **Have necessary and sufficient conditions**

Primitive classes

- **Have only necessary conditions**
  - **The necessary and sufficient space is empty**

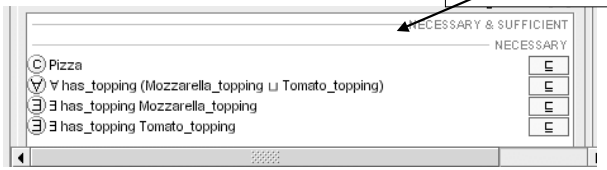
Defined

At least one Necessary & Sufficient condition

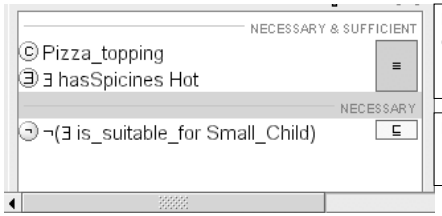


Primitive

No Necessary & Sufficient conditions



Defined classes with necessary conditions



Necessary & Sufficient conditions:

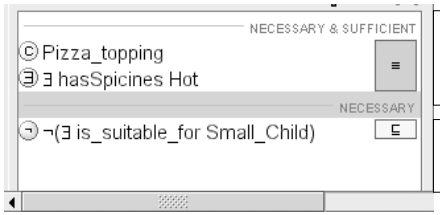
"Definition"

Necessary conditions:

"Description"

- **In effect this is a rule**
  - **IF Pizza_toping and hasSpiciness *some* Hot THEN not suitable_for *any* small_child**
    - **Easier to understand than separate subclass axioms.**

Protégé-OWL – Moving Conditions



Necessary & Sufficient conditions:

"Definition"

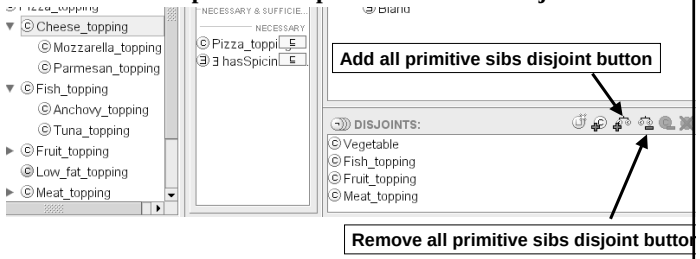
Necessary conditions:

"Description"

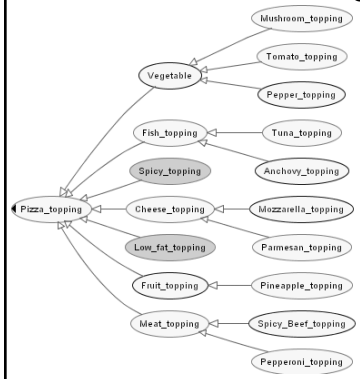
- **A common operation so:**
  - **Cut & Paste**
  - **Drag and Drop**
  - **One click – convert to/from defined/primitive class**

Managing Disjointness

- **Basic; Must be explicit; Easy to forget**
  So make it easy to do
  - **Disjoint primitive siblings button**
  - **"Create group of classes" Wizard**
  - **Annotate parent – all primitive children disjoint**



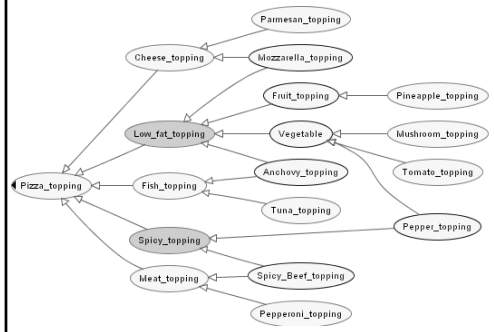
Understanding Classification



- **Asserted**
  - **Simple tree**
    - **Defined (orange) classes have no children**

Understanding classification

- **Inferred**
  - **Polyhierarchy**
    - **Defined (orange) classes have children**

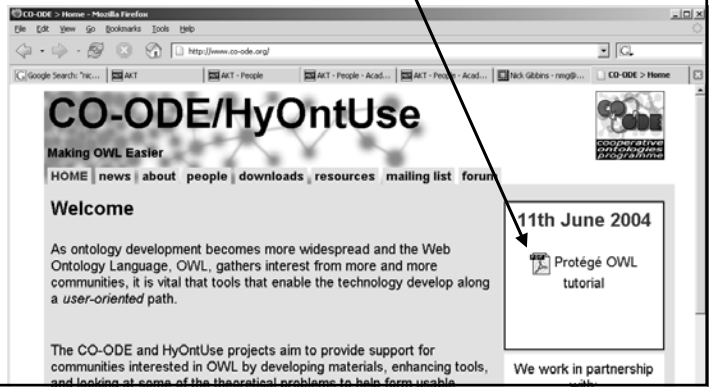


What to do when "Its all turned red"

Don't Panic!

- **Unsatisfiability propagates – so trace it to its source**
  - **Any class with an unsatisfiable filler in a *someValuesFor* condition is unsatisfiable**
  - **Any *subclass* of an unsatisfiable class is unsatisfiable**
- **Only a few possible sources**
  - **Violation of disjoint axioms**
  - **Unsatisfiable expressions**
    - **Confusion of "and" and "or"**
  - **Violation of a universal (allValuesFrom) constraint (including range and domain constraints)**
    - **Unsatisfiable domain or range constraints**
- **Tools coming RSN**

Web Site version
120 pp "Text book style"
www.co-ode.org



What's it Mean?

- **Paraphrases help clarify meaning**
  - **someValuesFrom** **"some"**
  - **allValuesFrom** **"only"**
  - **complete** **"A … is any … that…"**
  - **partial** **"All … are…have…"**
  - **negation** **"does not have … any…"**
  - **intersection** **"and" / "and also"**
  - **union** **"or" / "and/or"**
  - **not…someValuesFrom** **"not…any"**
  - **not…allValuesfrom** **"does not …have only…"**
  - **open world** **"amongst other things"**

33