Overview of Ontology Technology

from "Handbook on Ontologies"
by Steffen Staab, Rudi Studer (editors)

Buzzwords

An ontology is a **formal explicit specification** of a shared **conceptualization** for a **domain** of interest

Dictionary.com

- the metaphysical study of the nature of being and existence.
- (AI) An explicit formal specification of how to **represent** the **objects**, **concepts** and other **entities** that are assumed to exist in some area of interest and the **relationships** that hold among them.
- (information science) The hierarchical structuring of knowledge about things by subcategorising them according to their essential (or at least relevant and/or cognitive) qualities.

Outline

- Ontology Applications
- Ontology Representation & Reasoning
- Ontology Engineering
- Using Ontology

Applications

- Bioinformatics
- Semantic Web Browser
- Knowledge Managements

Common vocabulary

- Gene Ontology
- MGED standards of microarray data
  - Minimum Information About a Microarray Experiment (MIAME)
  - MicroArray and Gene Expression (MAGE)
- Unified Medical Language System

Schema Definition

- Frame-based representation (like OO)
  - Frame = class; Slot = Attribute | Relationship
- EcoCyc, MetaCyc, HumanCyc
- RiboWeb: the entire ribosome of prokaryotes
- PharmGKB: genetic and clinical data
- Reactome: pathways and reactions
- BioPAX: common language, pathway databases

Query Formulation with TAMBIS

- Transparent Access to Multiple Bioinformatics Information Sources
- Using Description Logic called GRAIL
  - have constraints on relationships (to deal with biological exceptions)
- One ontology shared by all resources
- Queries more complex than database query
  - new concepts can be defined dynamically with DL

Semantic Web Filter MagPie

- making sense of web pages
- “browsing” ontologies and KB when browsing web pages
- viewpoint of ontology, like view of database
- on-demand and triggered services

http://kmi.open.ac.uk/projects/magpie/main.html#movies

Knowledge Management

- Goal
  - Conversion of data & text into knowledge
  - Conversion human knowledge into machine form
  - Connection of people and knowledge to others
  - Creation of new knowledge
- Applications
  - Knowledge Portals
  - Organizational Memories
  - Expert Finder and Skill Management

Knowledge Portals

- Communities of Practice: informal, self-organizing group of individuals interested in a particular practice
- OntoWeb: http://www.ontoweb.org/
- KM portal: http://www.brint.com/
- RiboWeb:

  http://smi-web.stanford.edu/projects/helix/riboweb.html

Organizational Memories

- Knowledge & information of organizations
  - documents of best practice & lessons
  - continuous news articles
  - document templates
  - company regulations & manuals
  - minutes of meetings
  - human resource database

Expert Finder & Skill Management

- Expert finder
  - Yellow page systems
  - Expert directories
  - Personal web pages
- Skill management
  - SwissLife: customer management, insurance
  - OntoProper: profiles of skills of employees and applicants

Ontology languages

Requirements:

- well-defined syntax
- well-defined semantics
- efficient reasoning support
- sufficient expressive power
- convenience of expression

Ontology languages

- Description Logics
- F-logic
- Resource Description Framework
- Web Ontology Language

Typical reasoning

- Class membership

  x is an instance of C, and C is subclass of D then x is an instance of D

- Equivalence of classes
- Consistency: empty class should have no instance
- Classification: x is an instance of A

Description Logics

A man that is married to a doctor and has at least 5 children, all of whom are professors

$$\text{Human} \cap (\neg\text{Female}) \cap (\exists \text{married}.\text{Doctor}) \cap (\geq 5\_\text{has}\_\text{Child}) \cap (\forall \text{hasChild}.\text{Professor})$$

F-logic

man::person.

person(X) :- man(X)

person[father => man].

man(X) :- father(X,Y), person(Y)

FORALL X,Y X[son->> Y] <- Y:man[father-> X].

son(Y,X) :- man(Y), father(X,Y)

cain:man. eva:woman.

cain:man[father->adam; mother->eva].

FORALL X,Y <- X:woman[son->>Y[father->adam]

FORALL X,Y <- X:woman AND X[son->>Y] AND Y[father->adam]

Methods

- Parameters

  lisa:woman

  FORALL X <- adam[son@(lisa) ->> {X}]

  adam[son@{eva,1}->> cain]

  adam[son@{eva,2}->> abel]

- Overloading

  FORALL X <- adam[son->>X]

  (asking for all the sons of Adam)

Class and Subclass

adam:man.

eva:woman.

man::person.

woman::person.

class and subclass membership is partial-ordered

- Predicate symbols
- Name spaces for different ontologies
- Path expression

  FORALL X,Y <- X:woman[son->>Y[father->adam]

  FORALL X <- X.son[father->adam]
- Rules: AND, OR, NOT
- Allow expressive forms of negation
  - well-founded semantics
- Non-monotonic inheritance
  - class membership implicated by rules

Inheritance examples

c1 :: c2.

c2[m->>a].

c2[m->>b] <- c1[m->>a]

c1 :: c2.

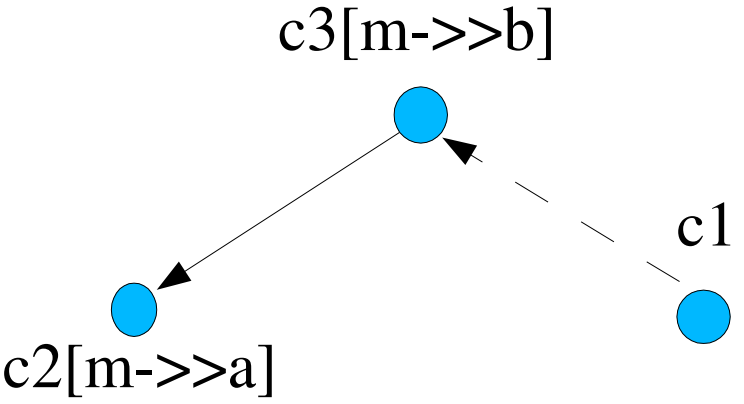
c2[m->>a].

c1[m->>b] <- c1[m->>a]

c1 :: c2. c3 :: c2.

c2[m->>a]. c3[m->>b].

c1::c3 <- c1[m->>a]



Ontobroker (Florid, Flora, etc ...)

C[A=>R]	atttype_(A,A,R)
C[A=>>R]	setatttype_(C,A,R)
C[A@(B1,...)=>R]	atttype_(C,A(B1,...),R)
A::B	sub_(A,B)
o:C	isa_(o,C)
o[A->b]	att_(o,A,b)
o[A->>b]	setatt_(o,A,b)
p(b1,...)	p(b1,...)

Additional rules to capture semantics

closure rules for X::Y

FORALL X,Y,Z sub_(X,Z) <- sub_(X,Y) and sub_(Y,Z)

attributes are inherited

FORALL C1,C2,A,T atttype_(C1,A,T) <- sub_(C1,C2) and atttype_(C2,A,T)

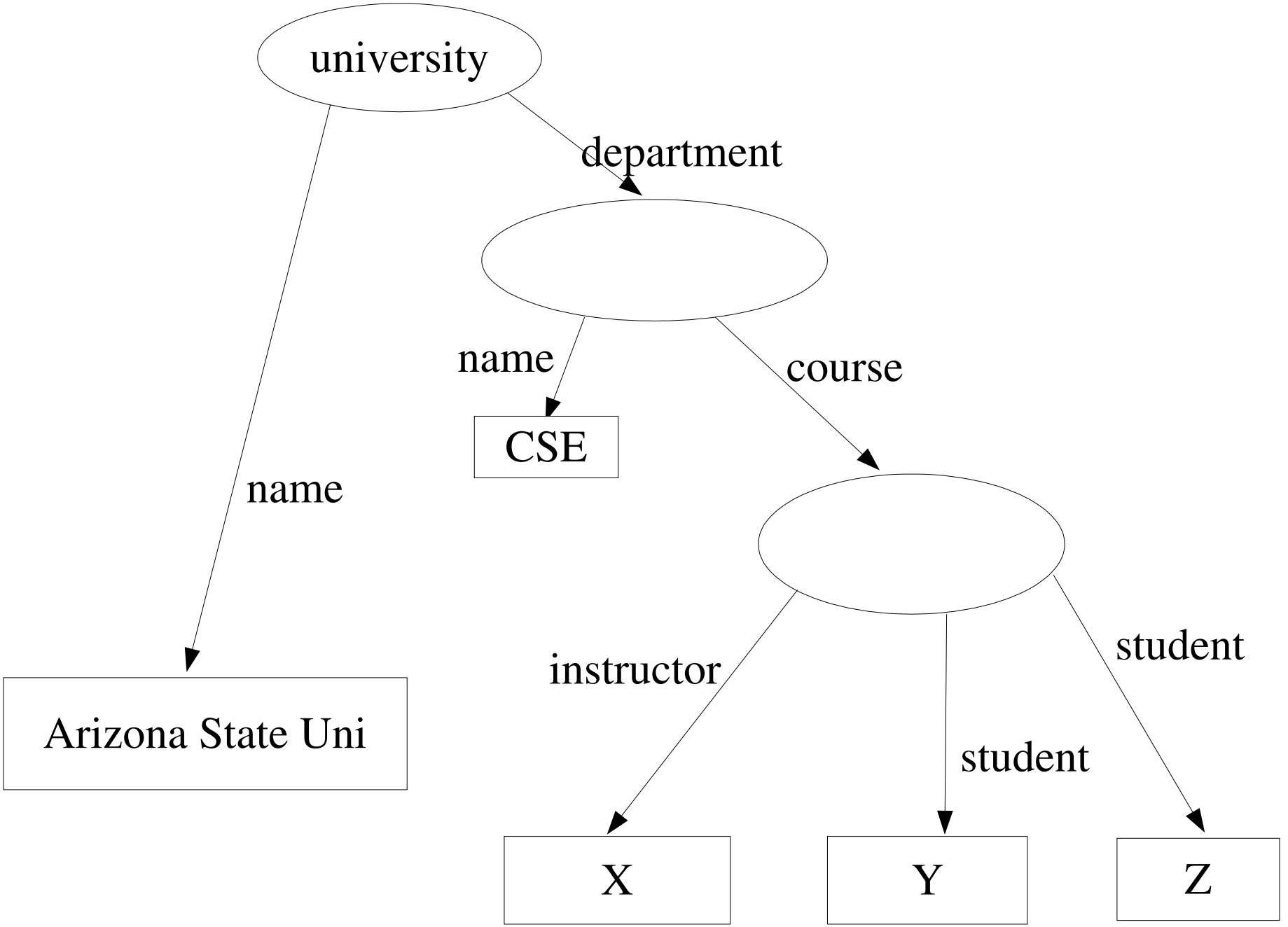
FORALL C1,C2,A,T setattype(C1,A,T) <- sub(C1,C2) and setattype(C2,A,T)

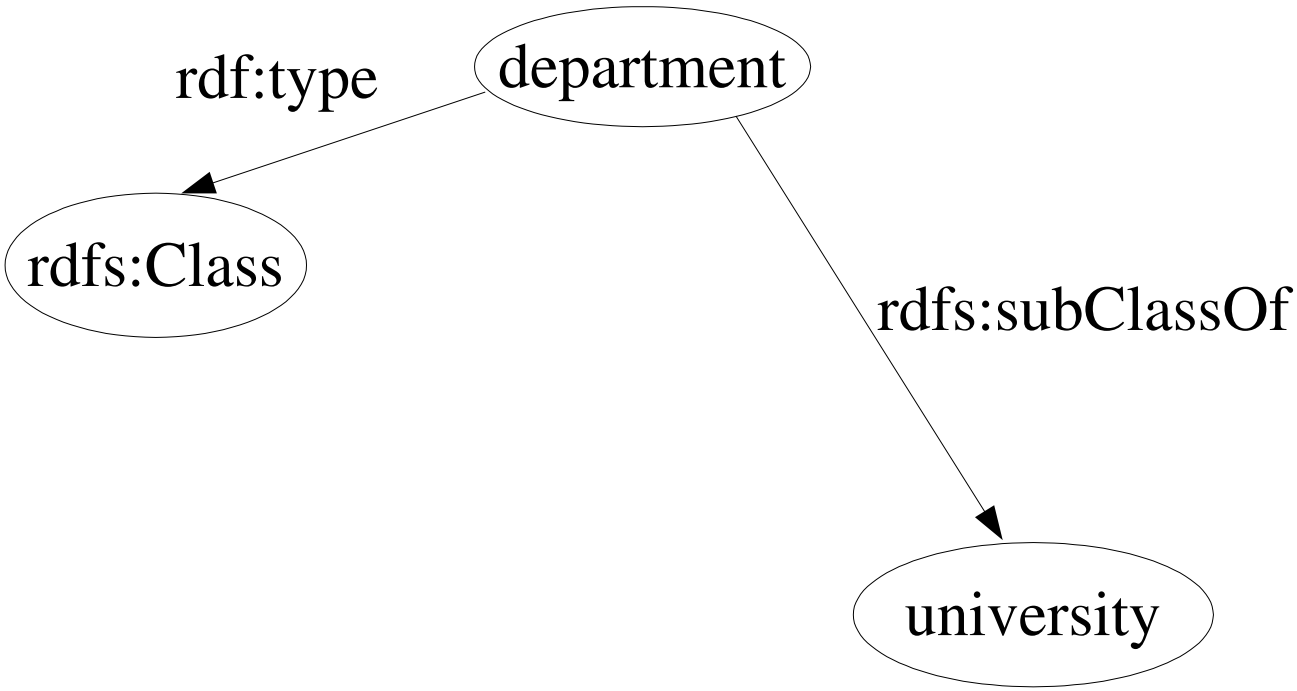
closure rules for X:C

FORALL O,C,C1 isa_(O,C) <- sub_(C1,C) and isa_(O,C1)

Resource Description Framework

- Originally designed for describing Web pages
- Resource = Identity (~ Object)
- An identity has properties (~Attribute/Method)
- RDF = instance; RDFS = schema
- Frame based, OO: resource-centric
- RDF(S): property-centric
- NOT supported:
  - equivalence, inverse relations, cardinality constraints
  - negation


university
department
name
course
name
CSE
name
Arizona State Uni
instructor
student
student
X
Y
Z



classes & subclasses
properties & subproperties
domain & range
data types
built-in data types

Web Ontology Language

- Extending RDF
- Trade-off:
  - expressive power
  - efficient reasoning
  - compatibility with RDF
- OWL Full: expressive, non-decidable, compatible
- OWL DL ~ Description Logic
- OWL Lite: very restricted expressive power

New features

- Constraints on properties
  - a person has exactly 2 parents
  - a course is taught by at least 1 lecturer
- Equivalence, disjointness of classes
- Union, intersection, complement of classes
- Characterizing properties:
  - *transitive*: '>'
  - *unique*, *inverse*

OWL example

```
<owl:Class  rdf:about="firstYearCourse">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty  rdf:resource="isTaughtBy"/>
      <owl:allValuesFrom  rdf:resource="#Professor"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Ontology Engineering

- Construction of large ontologies
  - Convert UMLS to Description Logic
- Ontology Learning
  - input: dictionaries, texts, semi-structure documents
  - clustering, classification, pattern mining (syntactic)
- Ontology Engineering Environments
  - browse, create, edit, modify
  - Ontolingua, Protege

Using/managing Ontology

- Extraction of knowledge patterns
  - Generalization, Induction
  - Modules, Motifs
  - Reuse
- Mapping/matching
- Merging
- Evaluation

Conclusion

- Goal
  - Common language
  - Independence of resources
  - Collaboration, Integration, Sharing
  - Efficiency
- State-of-the-art
  - Ontology ~ “universal database schema”.
  - Reasoning: DL, F-logic

What we can do?

- Study technical problems of ontology in LP
  - inheritance
  - induction of knowledge patterns
- Address engineering issues:
  - efficiency (OWL)
  - objects, functions (F-logic)
  - reuse (UMLS -> DL): Cyc -> AnsProlog?
- Reasoning plugin for Protege