Basics of ontology modelling

Knowledge Architecture

School of Computing
Napier University, Edinburgh, UK
Uta Priss

Semester 1, 2005

Outline

A dilemma for knowledge representation:

- unlimited opportunity (powerful hardware, software, Moore's law, graphics)
- but often: faulty systems, imperfect usability, confusing information architecture, high learning curve
- reasons: too short software life-cycle? interdisciplinarity? interoperability? lack of theoretical understanding?

What are the theoretical foundations of knowledge representation?

TBA

Why develop ontologies?

- Share common understanding of information among people or agents
- Reuse of domain knowledge
- Make domain assumptions explicit
- Separate domain knowledge from operational knowledge
- Analyse domain knowledge

(Note: the remaining part of this lecture is based on the Noy & McGuiness (2001) paper.)

What is an ontology

- classes (or concepts)
- relations (a subset of classes)
- slots (features, attributes, roles or properties)
- values with restrictions (facets), cardinality, type, scope
- instances (individuals, objects or entities)

$\longrightarrow$ similar to object-oriented modelling, relational databases, library thesauri.

Examples: An ontology for dogs

- classes: dog, poodle, terrier, ...
- slots: fur colour, size, ...
- value restrictions: size is between 30 cm and 1 m, ...
- instances: Greyfriar's Bobby

Some rules for creating ontologies:

- There is no one single correct way for building an ontology.
- Ontology development is iterative.
- Concepts of the ontology should be close to objects and relationships in the domain of interest.

$\longrightarrow$ rules are similar to those for entity relationship modelling in relational databases and for (library) thesaurus construction.

Class hierarchy

- Poodle is a subclass of Dog.
- Toy-Poodle is a subclass of Poodle.

All poodles are dogs.
The class Poodle has the same slots as Dog, but it can have additional ones.

Slots

Slots can be

- intrinsic properties (such as fur colour for dogs).
- extrinsic properties (such as region of origin).
- parts
- relations with other individuals (such as dog/breeder).

Constructing a class hierarchy:

- should be transitive
- avoid cycles
- create a list of synonyms for each class
- preserve level of granularity
- use neither too few, nor too many subclasses for each class
- multiple inheritance is ok
- there is no clear rule for deciding whether something is a class, a property value or an instance
- rules for naming (capitalisation, singular/plural, prefix/suffix) should be established