

# INFS3101 / 7100

## Ontology and the Semantic Web

### Review

## Examination

- ❖ Closed book
- ❖ 2 Hours working
- ❖ 10 minutes perusal
- ❖ Marked out of 60
- ❖ Ten questions, 6 marks each
- ❖ Questions have 3 or so parts each
- ❖ Short answers, point form ok
- ❖ Consultation
  - 7 June (Wednesday)
  - 16 June (Friday)

## Outline of Course

What is going on in interoperating systems.
Speech acts and institutional facts.
Semantic heterogeneity
Complex objects - the part/whole relationship
Complex structures - subclasses and subproperties
Formal upper ontologies
Quality of ontology – Gruber's principles
Semantic web view – RDFS
Semantic web view – OWL
Advanced issues
Predicates

## What is going on in interoperating systems

- ❖ For **agents to interoperate** on the **semantic web**, they need an **ontology** supporting an **exchange of trusted sources**.
- ❖ The ontology includes **message types, message contents, and descriptors of services**.
- ❖ An ontology is a **world**, so must include **individuals** as well as **types**.

## Speech acts and institutional facts.

- ❖ **Institutional fact** is a record of a **speech act**. **Brute fact X** counts as institutional fact **Y** in **context C**.
- ❖ Context includes **framing rules and background**.
- ❖ Most information systems manage records of speech acts. Interoperating agents perform speech acts, both **performative and informative**.

## Semantic heterogeneity

- ❖ **Bottom-up approach** to building an ontology fails, due to **semantic heterogeneity**. To interoperate, organisations need to align their business practices, implemented as systems of speech acts. Requires integrating contexts, including background.
- ❖ The ontology is a record of agreement created prior to the interoperation. Participating organisations must **commit** to the ontology. Approach is **top-down**.
- ❖ A system of interoperating agents operates within a single system of speech acts.

## Complex objects - the part/whole relationship

- ❖ Complex objects need both **identity** and **unity**. Identification and unification can be **logical** or **lexical**. **Independent** classes can help identify **dependent** classes. A whole can be identified **metonymically**. Identity and unity depend on **context**. Can be **indexical**.
- ❖ **Countable** types have both identity and unity. **Bulk** types lack one. **Containers** can give **pseudo-identity** to a bulk type.

## Complex structures - subclasses and subproperties

- ❖ Common to represent things as **objects in classes**. Objects have **properties**. Superclass **subsumes** subclass, superproperty **subsumes** subproperty. Properties have **metaproperties** **rigid**, **essential**, **identity**, **unity** which govern subsumption. Subclasses can be **defined** or **declared**.

## Formal upper ontologies

- ❖ **BWW** and **Dolce** are **formal upper ontologies**. Include **systems**, **endurants**, **perdurants**, **subclasses**, etc. Formal upper ontologies provide a rich meta vocabulary to help develop ontologies and suggest **abstract data types** which can be supported by ontology servers to make it easier to build rich ontologies.

## Quality of ontology – Gruber's principles

- ❖ Quality principles for ontologies include **clarity**, **coherence**, **extendibility**, **encoding bias** and **ontological commitment**. Want to maximise the first three and minimise the last two.
- ❖ Quality involves **cost-benefit tradeoffs**.

## RDFS

- ❖ **RDF** elaboration of hyperlink as **triple**, **resources** linking resources in **namespaces**. Allows **literals** and **blank nodes**. **RDFS** allows **class**, **subclass**, **subproperty**, **domain**, **range**. But not structure comparable to **UML**. Need **OWL** for that.

## OWL

- ❖ **OWL** extends **RDFS**. **Object** and **datatype properties**. Universal class **Thing**. **Restrictions** are subsets of property domains. **Names** not satisfy unique names assumption. Class descriptions can be **enumeration** or **boolean combination** of classes. **Ontology properties**. **OWL Full**, **OWL DL**, **OWL Lite**.

## Advanced issues

- ❖ Widely-used structural features include: **countable/ bulk classes, concept/representation classes, dimension systems, mereological structures, metaproperties and extent-descriptive metaclasses.** All can be modelled using extensions to OWL.

## Predicates

- ❖ **Common logic** is a language to express **predicates**. Can map OWL Full to CL. But CL is richer than OWL Full, so can be used to extend OWL. But need to be careful, since CL open to paradox.
- ❖ Need to be able to read CL predicates, not necessarily write them.