OWL General Requirements

Jeff Heflin
Committee

- Jeff Heflin (co-chair)
- Deborah McGuinness (co-chair)
- Jeremy Carroll
- Dan Connolly
- Jos De Roo
- Pay Hayes
- Ned Smith
- Herman ter Horst
Format

• Name and number
• Supported tasks
  – What does requirement allow us to do?
• Justification
  – Why is this requirement needed?
• Possible approach
  – How can the language support the requirement?
• DAML support
  – To what extent does DAML+OIL support it?
What is a “Requirement?”

• Possible criteria
  – “if we don’t meet it, we aren’t done” – Dan C.
  – must result in language primitives
  – must be implemented in all OWL systems?
  – appropriate for the “ontology layer” of the Semantic Web?
  – critical for some very important use cases?
  – …

• Some of these are debatable!
R1. Shared Ontologies

- Ontologies are publicly available and different data sources can commit to the same ontology for shared meaning.

Possible Approach:
- Syntax for defining ontologies
- Syntax for committing to ontologies
- Syntax for disambiguating terms from different ontologies
R2. Ontology Extension

- Ontologies can be extended by other ontologies in order to provide additional definitions
- Possible Approach:
  - Explicit representation of extension
R3. Ontology Evolution

• Ontologies can be changed over time and data sources can specify which version of the ontology they commit to.

• Possible Approach:
  – Revisions are separate documents
  – Explicit links to prior versions
  – Explicit backwards-compatibility
  – Deprecation of terms
R4. Ontology Interoperability

• Different ontologies may model the same concepts in different ways

• Possible Approach:
  – primitives for mapping
  – consider some of (but not all) the following
    • subclass/superclass
    • inverses
    • equivalence
    • implication, arithmetic, aggregation, string manipulation, procedural attachments?
R5. Detect Inconsistency

- Different ontologies or data sources may be contradictory
- Possible Approach:
  - allow language to express inconsistency
  - theory supports efficient detection of inconsistency
  - provide mechanism for reporting inconsistencies
R6. Scalability

- Language can be used with large ontologies and large data sets
- Must balance with R10. Expressiveness
- Possible Approach:
  - restrict language for efficient reasoning
    - description logic
    - datalog
R7. Ease of Use

• Language should provide a low learning barrier and have clear concepts and meaning

• Possible Approach:
  – When possible, use concepts and idioms familiar to average software engineers
    • object-oriented?
    • relational databases?
R8. XML Syntax

• The language should have an XML serialization

• Open Issue:
  – Must the language also build on RDF/RDFS?
    • In favor of RDF
      – W3C standard
      – Existing software support
    • Against RDF
      – Does not have same acceptance as XML
      – Led to an awkward syntax for DAML+OIL
R9. Ontology-based Search

- Search that exploits the meaning of terms instead of just the syntax
- Possible Approach:
  - use background ontologies for:
    - query expansion
    - understanding of term relationships
    - identify parameters and value restrictions
R10. Expressiveness

- The language should be as expressive as possible, given a balance with R6.
  Scalability

- Should probably combine this with R6 for:
  - Balance of Expressiveness and Scalability
Other candidates (Goals?)

- C1. Explainability
- C2. Internationalization
- C3. Ontology querying
- C4. Tagging
- C5. Proof checking
- C6. Security
- C7. Trust
- C8. Data persistence