Ontology Engineer Requirements

Focus on what ontologists need to DO and KNOW

Dr. Michael Uschold
Independent Consultant
LinkedIn Profile

Customers and Stakeholders

Ability to interview customers and stakeholders

- Determine ontology requirements
- Determine whether an ontology-based solution is appropriate
- Play devil's advocate, if they insist they need an ontology but cannot articulate why, then tell them they don't need one.

Competency Questions

Ability to

- Formulate ontology requirements as competency questions
- Informal competency questions first
- Include formal vocabulary
- Formal competency questions
- Drives the ontology engineering process

Avoid Terminology Wars

- Know how to avoid terminology wars for ambiguous terms like “process” “product” or even “architecture”.

  RECIPE:

1. Prohibit use of the term in question, make up terms like foo1, foo2.
2. Gather all the different definitions:
   - foo1: agreed definition of meaning 1
   - foo2: agreed definition of meaning 2
   - ...
   - fooN: agreed definition of meaning N

RECIPE: Avoid Terminology Wars

3. Identify common elements of each definition
4. Agree on one or more meanings that you want to have terms for
5. Decide on what the terms will be
6. Fight about the terms only *after* you know exactly what you are talking about.

Languages and Tools

Be familiar with wide range of languages and tools

- OWL, RDF, FOL, CL, Flogic
- Inference engines: Pellet, Fact++, Ontobroker, KAON2, ...
- Semantic computing infrastructure
  - Triple stores
  - Open source platforms
- See: www.mkbergman.com
  Sweet Compendium of Ontology Building Tools

Languages and Tools

- Identify candidate languages and tools that can meet customer requirements
- Know key criteria informing a choice
  - performance, conformance to standards, expressivity, maintainability
- Evaluate languages/tools according to criteria
  - pros/cons as apply to customer's context
- Make recommendation taking tradeoffs into account

Comprehensive Tools

- **Altova SemanticWorks** is a visual RDF and OWL editor that auto-generates RDF/XML or nTriples
- **Amine** is a rather comprehensive, open source platform for the development of intelligent and multi-agent systems written in Java.
- **The Apelon DTS (Distributed Terminology System)** is an integrated set of open source components that provides comprehensive terminology services in distributed application environments.
- **DOME** is a programmable XML editor which is being used in a knowledge extraction role to transform Web pages into RDF, and available as Eclipse plug-ins.

Comprehensive Tools

- **FlexViz** is a Flex-based, Protégé-like client-side ontology creation, management and viewing tool; very impressive.
- **Knoodl** facilitates community-oriented development of OWL based ontologies and RDF knowledge bases. It also serves as a semantic technology platform, offering a Java service-based interface or a SPARQL-based interface so that communities can build their own semantic applications using their ontologies and knowledgebases.
- **ontopia** is a relative complete suite of tools for building, maintaining, and deploying Topic Maps-based applications; open source, and written in Java.

Comprehensive Tools

- **Protégé** is a free, open source visual ontology editor and knowledge-base framework.
- **TopBraid Composer** is an enterprise-class modeling environment for developing Semantic Web ontologies and building semantic applications. Fully compliant with W3C standards,

Other Tools

- Initial ontology development tools
  - concept map tools
  - ontology learning tools
  - importing from other formats, spreadsheets, databases etc.
- Ontology Editors
- Ontology Mapping
- Ontology Visualization and Analysis tools

Importance of names

- Understand and apply good practice for naming ontologies, classes, relationships and instances
- URIs and versioning a particular problem
  - avoid proliferation of multiple URIs for exactly the same thing
  - avoid changing the meaning of an existing URI it is has been publicly available / sanctioned
  - Example: SKOS experience
  - See: A URI Crisis

SKOS URI Challenge

- SKOS dilemma, minting URIs for new version
- There are no guidelines to work from
  - Change semantics and keep URI same?
  - Same semantics, mint a new URI?
  - One ontology, two namespaces?
- URIs are overloaded
  - organization, dates, version number,
  - file structure, meaning of concept
- Versioning not solved

Modular Design

- Know how to design the architecture for an ontology, identifying different modules and how they relate
- Known how to find and reuse existing ontologies rather than reinvent the wheeel
- Know when to create a separate ontology module that can be reused vs. creating special purpose components

Design Patterns

- Be familiar with ontology design patterns
  - Catalogue: http://www.ontologydesignpatterns.org
- Use them whenever possible
- Create new ones and publish to community

Standards

- Be familiar with all relevant standards
- Know when and how to apply them

Resources

- Know how to find and use resources
- Books
- Ontology portals, academic and commercial
- Communities of practice

FUN

- KNOW how to have fun being an ontology professional
- DO have fun!