Ontology Engineering

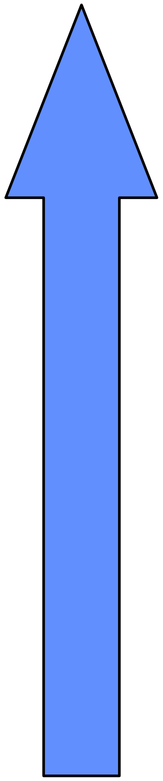
Mark S. Fox
University of Toronto
msf@eil.utoronto.ca

Mark Musen
Stanford University
musen@stanford.edu

Readings

- Noy, N.F., and McGuinness, D.L., (2001), “Ontology Development 101: A Guide to Creating Your First Ontology”, Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880.
- [Optional] Grüninger, M., and Fox, M.S., (1995), “Methodology for the Design and Evaluation of Ontologies”, Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, IJCAI-95, Montreal.

Ontology Hierarchy



- Protégé
  - A graphical user interface for defining classes and properties using the OWL/DL semantics
- OWL
  - An implementation of Description Logic conforming to Semantic Web/Linked Data principles
- Description Logic
  - A logical language for defining classes and properties

Design Options

- There may be several ways of representing any given problem, and we often need to search through these different possibilities.
- What are the possible alternatives?
- A declarative specification of a system provides a precise and rigorous characterization of the space of these alternatives -- if it's consistent with the specification of the ontology, then it is a possible alternative model.

Ontology Design Criteria

- **Competence**: What questions can the representation answer or what tasks can it support?
- **Generality**: To what degree is the representation shared between diverse activities?
- **Granularity**: Does the representation support reasoning at various levels of abstraction and detail?
- **Perspicuity**: Is the representation easily understood by the users?

Ontology Design Criteria (after Gruber)

- **Clarity**: Definitions should be objective and complete
- **Coherence**: The ontology should sanction those inferences consistent with the definitions
- **Extendibility**: An ontology should anticipate future uses
- **Minimal encoding bias**: No assumptions about knowledge representation
- **Minimal ontological commitment:** Say no more than is necessary

Trade-offs in Ontology Design

- Minimizing ontological commitment requires specifying a weak theory
- Making definitions precise requires increasing ontological commitment
- Anticipating various uses of the ontology may require increasing the number of concepts represented
- Making an ontology maximally general may make it useless for any specific application

Different Philosophies for Scoping Ontologies

- Be as encyclopedic as possible: The more that can be modeled the better
  (cf. CYC, NCI Thesaurus)
- Let a thousand flowers bloom: Create small-scale ontologies, each tailored for a relatively few number of tasks

Ontology Engineering Steps

1. **Determine domain and scope**
2. Determine the Competency Questions
3. Consider reusing existing ontologies
4. Enumerate important terms
5. Identify classes and structure as a taxonomy
6. Define classes using properties
7. Define instances
8. Validate using competency questions

Determine domain and scope

1. What is the domain that the ontology will cover?
2. Who will use and maintain the ontology?
3. How will the ontology be used?
   - What types of questions should the ontology be able to answer?
   - What information will applications want to retrieve from it?

Course Assignment Problem

- Our example is drawn from academe, in particular it will support a department in deciding which courses are to be taught and by whom.
- The goal is to develop an ontology that can model both the courses to be taught, the skills required to teach them, and the staff from whom possible teachers are drawn.

Scenarios

- A scenario describes a specific persona and a specific task that persona performs using the system based on the ontology to be designed.

  "Jane is responsible for assigning courses to faculty for MIE. Her first task is to find out which courses each faculty member taught the prior year. She then identifies which faculty will be on leave or sabbatical the coming year. Based on who will be away, she identifies which courses are not covered. For each uncovered course, she searches for a faculty member in the area of the course that does not have a full teaching load …"

Ontology Engineering Steps

1. Determine domain and scope
2. **Determine the Competency Questions**
3. Consider reusing existing ontologies
4. Enumerate important terms
5. Identify classes and structure as a taxonomy
6. Define classes using properties
7. Define instances
8. Validate using competency questions

Competency

- Competency is defined by a set of questions the ontology is able to answer
  - Competency determines the scope of an ontology's question answering/information retrieval capabilities
  - In other words it determines what types of problem solving it can support

Ontology Competency ----Distance---- Task Competency Requirements

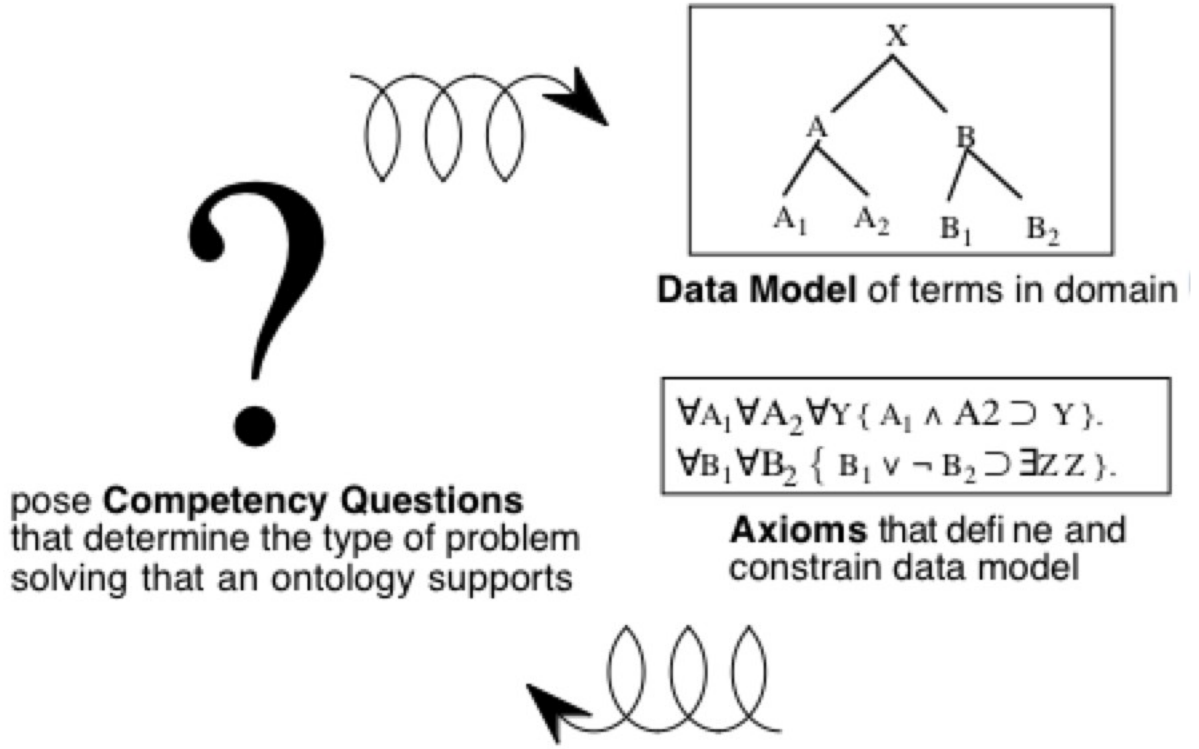
Want to choose/design an Ontology that minimizes the the distance between your task needs and the ontology's competency.

How to Develop Competency Questions

- Start with the simplest questions and then elaborate
  - **Instance Existence questions:**
    - Is there a professor named Mark Fox?
  - **Data Property questions**:
    - What is the age of the professor?
  - **Object Property questions:**
    - Does Mark teach MIE1501?
  - **Classification questions:**
    - What types of professors are there?
  - **Reasoning questions**:
    - Can I schedule MIE1501F at 9am in BA1420?

Questions Requiring Reasoning

- The ontology must contain the necessary axioms to solve competency questions requiring reasoning.


X
A
B
A_1
A_2
B_1
B_2
Data Model of terms in domain
$\forall A_1 \forall A_2 \forall Y \{ A_1 \wedge A2 \supset Y \}.$
$\forall B_1 \forall B_2 \{ B_1 \vee \neg B_2 \supset \exists Z\, Z \}.$
pose **Competency Questions** that determine the type of problem solving that an ontology supports
Axioms that defi ne and constrain data model

Course Assignment Competency Questions

- What courses does MIE offer?
- What skills are required to teach course x?
- Who of the faculty possesses those skills?
- Who of the faculty can teach course x?
- What courses did faculty member x teach in year y?
- *What did we leave out?*

Ontology Engineering Steps

1. Determine domain and scope
2. Determine the Competency Questions
3. **Consider reusing existing ontologies**
4. Enumerate important terms
5. Identify classes and structure as a taxonomy
6. Define classes using properties
7. Define instances
8. Validate using competency questions

Consider Reusing Existing Ontologies

- There may already exist ontologies that meet your needs
- Your application may have to interact with other applications that are using an ontology
- In either case, use the competency questions for each ontology, if available, to determine whether they are adequate for your task
  - Is there a corresponding competency question in the other ontology that matches your competency question

Example

- Google search for “university ontology” returns
  - http://www.cs.umd.edu/projects/plus/SHOE/onts/univ1.0.html
  - Malviya, N., Mishra, N., and Sahu, S., (2011), “Developing University Ontology using protégé OWL Tool: Process and Reasoning”, *International Journal of Scientific & Engineering Research*, Vol. 2, No. 9, pp. 1-8. http://www.ijser.org/researchpaper%5CDeveloping_University_Ontology.pdf
- Other relevant ontologies
  - Fox, M.S., Barbuceanu, M., Gruninger, M., and Lin, J., (1998), “An Organisation Ontology for Enterprise Modeling”, In *Simulating Organizations: Computational Models of Institutions and Groups*, M. Prietula, K. Carley & L. Gasser (Eds), Menlo Park CA: AAAI/MIT Press, pp. 131-152. http://www.eil.utoronto.ca/wp-content/uploads/enterprise-modelling/papers/org-prietula-23aug97.pdf
    - Ontology: http://ontology.eil.utoronto.ca/organization.owl

CMU University Ontology

class Hierarchy

```
Person*
        Employee*
            Faculty
                Professor
                    AssistantProfessor
                    AssociateProfessor
                    FullProfessor
                    VisitingProfessor
                Lecturer
                PostDoc
            Assistant
                ResearchAssistant
                 TeachingAssistant
            AdministrativeStaff
                 Director
                Chair {Professor}
                Dean  {Professor}
                 ClericalStaff
                SystemsStaff
        Student
```

This is only the terminology portion of an ontology. No semantics are provided.

Therefore, it is not a complete ontology! Even the competency questions are missing!

CMU University Ontology

```
Organization*
      EducationOrganization*
         Department
         Institute
         Program
         ResearchGroup
         School
         University
```

CMU University Ontology

```
Publication*
      Article*
         BookArticle*
         ConferencePaper*
         JournalArticle*
      WorkshopPaper*
      Book*
      Periodical*
         Journal*
         Magazine*
      Proceedings*
      Thesis*
         DoctoralThesis*
         MastersThesis*
```

Ontology Engineering Steps

1. Determine domain and scope
2. Determine the Competency Questions
3. Consider reusing existing ontologies
4. **Enumerate important terms**
5. Identify classes and structure as a taxonomy
6. Define classes using properties
7. Define instances
8. Validate using competency questions

Enumerate important terms

- Before getting too structured, write down all of the classes and properties that come to mind, i.e., a stream of consciousness approach
- Don't worry if they overlap or are inconsistent

Example

- Course, taughtBy, requiresSkill, forYear
- Professor, hasTaught, hasSkill
- Skill
- Program, hasCourse
- Department, hasProgram
- Faculty, hasDepartment
- University, hasFaculty

Ontology Engineering Steps

1. Determine domain and scope
2. Determine the Competency Questions
3. Consider reusing existing ontologies
4. Enumerate important terms
5. **Identify classes and structure as a taxonomy**
6. Define classes using properties
7. Define instances
8. Validate using competency questions

Naming Convention

- Define a naming convention and stick to it
  - **Classes**: Professor, MIEProfessor
    - No spaces, capitalize each word in the name
  - **Properties**: hasProperty
    - Use hasX to denote that a class has the value of the property
    - No spaces
- You can either define classes directly in Protégé, or write them down using DL

Define Classes

- Define taxonomy of classes
  - **Top down** - from most general to most specific
  - **Bottom up** - from most specific to most general
  - **Combination** - start where you feel most comfortable (i.e., the most salient classes) and work up or down from there

Example

- Employee
  - AcademicEmployee ⊏ Employee
    - TeachingStaff ⊏ AcademicEmployee
      - Professor ⊏ TeachingStaff
        - FullProfessor ⊏ Professor
        - AssociateProfessor ⊏ Professor
        - AssistantProfessor ⊏ Professor
      - Lecturer ⊏ TeachingStaff
  - AdministrativeEmployee ⊏ Employee
    - FinanceOfficer ⊏ AdministrativeEmployee
    - AdministrativeAssistance ⊏ AdministrativeEmployee
    - Secretary ⊏ AdministrativeEmployee
- Course
  - UniversityCourse ⊏ Course
    - EngineeringCourse ⊏ UniversityCourse
      - MIECourse ⊏ EngineeringCourse

Example

- University $\sqsubset$ org:Organization
- Faculty $\sqsubset$ org:Division
- Department $\sqsubset$ org:Division
- Program
- sc:Person

- org: http://ontology.eil.utoronto.ca/organization.owl#
- sc: http://schema.org/

Ontology Engineering Steps

1. Determine domain and scope
2. Determine the Competency Questions
3. Consider reusing existing ontologies
4. Enumerate important terms
5. Identify classes and structure as a taxonomy
6. **Define classes using properties**
7. Define instances
8. Validate using competency questions

Define Properties

- Two types of properties: object and data
  - Need to declare each before using
- Two approaches to using properties
  - Specify domain and range independent of the classes they are used in –
    - hold over from RDF
    - Does not take advantage of DL's capability to define classes
  - Define property domain and range within the class it is used
    - Uses DL's class definition capabilities
    - Can define both primitive and definitional classes

Properties

- Object Properties
  - **Course**: taughtBy, requiresSkill
  - **Professor**: hasTaught, hasSkill
  - **University**: hasFaculty, hasDepartment, hasProgram, hasCourse
- Data Properties
  - forYear
    - Could use Owl Time ontology to specify, but would then be an object property

Primitive Concepts

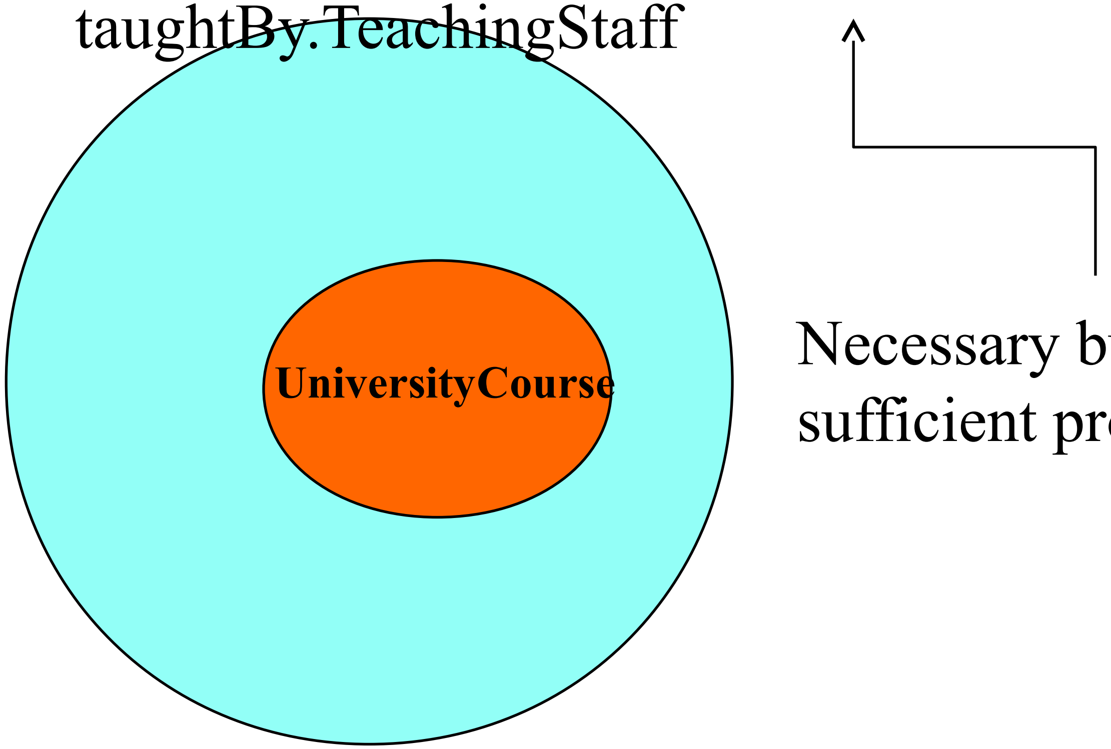
- Primitive concepts can only define the necessary conditions (i.e., properties) for individuals being a member of the concept

$$\text{UniversityCourse} \sqsubseteq \text{Course} \sqcap \geq 1\ \text{requiresSkill.Skill} \sqcap \geq 1\ \text{taughtBy.TeachingStaff}$$

- UniversityCourse is a primitive concept. It states that it is necessary that
  - it is a Course,
  - it has at least 1 requiresSkill property whose value is Skill, and
  - at least 1 taughtBy property whose value is TeachingStaff.

Primitive Concept

Course $\sqcap \geq 1$requiresSkill.Skill $\sqcap \geq 1$ taughtBy.TeachingStaff



Necessary but not sufficient properties

Defined Concepts

- Defined concepts define both the necessary and sufficient conditions for individuals being a member of the concept.

$$\begin{aligned} \text{UniversityCourse} \equiv\ & \text{Course} \\ & \sqcap \geq 1\ \text{requiresSkill.Skill} \\ & \sqcap \geq 1\ \text{taughtBy.TeachingStaff} \end{aligned}$$

- UniversityCourse is a defined concept. It states that
  - (necessary) if you are a UniversityCourse, it is **necessary** that you are Course with at least 1 property requiresSkill whose value restricted to Skill, and at least 1 taughtBy property whose value is TeachingStaff, and
  - (sufficient) if you are Course with at least 1 property requiresSkill whose value restricted to Skill and at least 1 taughtBy property whose value is TeachingStaff, then you are a UniversityCourse.

Other Classes

- $\text{MIE1501} \sqsubseteq \text{UniversityCourse} \sqcap {=}1\ \text{requiresSkill}.\{\text{artificialIntelligence}\}$

- How is the organization structure of a University represented?

Ontology Engineering Steps

1. Determine domain and scope
2. Determine the Competency Questions
3. Consider reusing existing ontologies
4. Enumerate important terms
5. Identify classes and structure as a taxonomy
6. Define classes using properties
7. **Define instances**
8. Validate using competency questions

Define Instances (Individuals)

Skill(artificialIntelligence)

Department(mie)

Program(ie)

hasCourse(ie, mie1501-2016)

# Define a professor.

FullProfessor(markSFox)

inDepartment(markSFox, mie)

hasSkill(markSFox, artificialIntelligence)

hasTaught(markSFox, mie1501-2016)

# Define an instance of MIE1501 taught in 2016

Ontology Engineering Steps

1. Determine domain and scope
2. Determine the Competency Questions
3. Consider reusing existing ontologies
4. Enumerate important terms
5. Identify classes and structure as a taxonomy
6. Define classes using properties
7. Define instances
8. **Validate using competency questions**

Validate Using Competency Questions

- Rewrite competency questions into a query language such as SPARQL.
- Test that each question can be answered correctly using the classes, properties and instances defined earlier

Competency Question

- What courses does MIE offer?

  -
    ```
    SELECT ?course
    WHERE {mie, hasProgram, ?p .
               ?p, hasCourse, ?course }
    ```
- What skills are required to teach course x?
- Who of the faculty possesses those skills?
- Who of the faculty can teach course x?
- What courses did faculty member x teach in year y?

Tips

Ontology Engineering Pointers

- Distinguish between classes and individuals
  - mie is an individual, Department is a class
- Siblings should be similar in class
  - mieCourse vs eceCourse vs writingCourse
- Number of siblings should be 2-10, otherwise may need another subclass in-between
- Subclasses usually have an additional property, or a restriction on its range

Class Creation Heuristics

- If the classes with different property values become restrictions for different properties in other classes , then we should create a new class for the distinction. Otherwise, we represent the distinction in a property value
  - Is the skill required for a course reason enough to distinguish the course
- If a distinction is important in the domain and we think of the classes with different values for the distinction as different kinds of classes, then we should create a new classes for the distinction

Ontology Completion Heuristics

- The ontology should not contain all the possible information about the domain: you do not need to specialize (or generalize) more than you need for you application.
- The ontology should not contain all the possible properties of and distinctions among classes in the hierarchy.
- Generalize? Yes, but do not over do it.