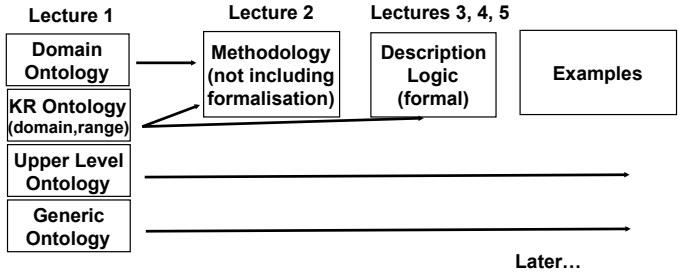
Ontologies

- Road map of next few lectures



Ontologies

- Symbolic representation of knowledge has a long history in AI
  - lisp: (list `ant `bat `cat)
  - prolog: [ant, cat, bat]
    mammal(X) :- cat(X).
- Ontologies establish standards and conventions for domain knowledge
  - Modelling: class, instance, relation
  - Content: objects vs processes,
  - Language: FOL, Description Logic
  - Definitional not problem solving knowledge

Ontologies

Reminder

- Application Ontologies
  - Describe specific domains
    - » Travel & Tourism; Medicine; Genetics
  - Can have diverse uses
    - » KBS (e.g. diagnosis); recommendation; web site / database design; data mining; NLP
- Knowledge Representation Ontologies
  - Define the formal vocabulary that specifies classes and relations
  - subClassOf; type
  - domain; range; subPropertyOf

Ontology Development Methodology

- Method 1
  - 'Methontology' – see pdf files in ontology-reading.tar.gz

    Ontological Engineering. Gomez-Perez, A., Fernandez-Lopez, M., and Corcho, O. Springer 2004, pp.125-142

    Building a chemical ontology. Fernandez-Lopez, M., Gomez-Peres, A. and Pazos Sierra, A. *IEEE Intelligent Systems* Jan/Feb 1999

    Building legal ontologies. Corcho, O., Fernandez-Lopez, M., Gomez-Perez, A. and Lopez-Cima, A.
- Method 2
  - Protégé

    Ontology Development 101. Noy, N. and McGuinness, D. Stanford Technical Report
- Method 3
  - Ontology design patterns

Methontology

- Emphasises
  - Ontology building as a craft
    - » Considering Domain Ontologies cf KR or Generic Ontologies
  - Lack of tested and generalised methodologies
  - The developers move too quickly to implementation
    - » Conceptual models get encoded in the implementation and not made explicit
    - » Ontological commitments not made explicit
    - » Experts are unable to critique formal ontologies
    - » The ontology encoding language imposes a bias on what *can* be said, so developers do not consider what *ought* to be said

Methontology

- The answer - to build ontologies at the 'knowledge level', that is, at a level of abstraction above the encoding language
  - An encoding language might be FOL or Description Logic, Prolog, or Java
  - Recommend the use of intermediate abstractions, e.g. tables and diagrams, to capture the objects in the domain and the relationships between them
- Methontology identifies the steps in the process
  - Described as a life-cycle of evolving prototypes
  - Beginning with knowledge acquisition and
  - ending with a formal ontology

Methontology

The ontology evolves from a series of prototypes
- Cyclical process
- The ontology development process describes the activities to be performed during ontology engineering

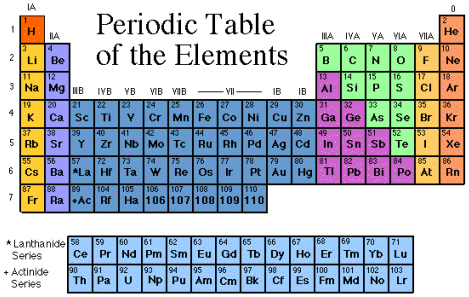
- Lifecycle of development activities
  - Specification: identify uses and users
  - Conceptualisation: structures the domain knowledge
  - Formalisation: transforms conceptual model into a formal model
  - Implementation: build the computable model
  - Maintenance: updates and corrects the ontology
- Support activities
  - Knowledge acquisition
    - » Predominantly at the beginning
  - Integration/Merging/Alignment
    - » Bring in existing ontologies
  - Evaluation
    - » Determine that modelling decisions are correct
  - Documentation

Methontology

- The Specification describes the
  - Primary objective
  - Purpose
  - Granularity level
  - Scope
- Conceptualisation
  - Organise the acquired knowledge
  - Represent it in a way both domain expert and ontologist can understand
    - » Intermediate representation - diagrams and tables
    - » 11 steps that can be performed iteratively

Methontology: Example

The chemical elements ontology, including all 103 elements and their properties. [Fernandez-Lopez]



Methontology: Example

The chemical elements ontology, including all 103 elements and their properties.

- **The Specification describes the**
  - **Primary objective**
    - » ***Build a reusable resource***
  - **Purpose**
    - » ***To be used when information about chemical elements is required in teaching, manufacturing etc***
  - **Granularity level**
    - » ***Elements, not compounds or sub-atomic particles***
  - **Scope**
    - » ***Cover the 103 elements, atomic no. and weight, electronegativity and melting point***
- **Knowledge acquisition**
  - **Preliminary meetings with experts to gain an overview**
  - **Study documents for familiarisation**
  - **KA with experts: (non)structured interviews; text analysis; reviews of intermediate representations**

Methontology: Example

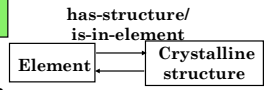
The chemical elements ontology

1. **Build the Glossary of terms**
   - **Consider anything relevant including concepts, instances, attributes, verbs**
2. **Build the Concept taxonomy**
   - **Organise the terms that represent concepts (i.e. classes or sets) into a concept hierarchy, distinguish:**
     - **subClassOf**
     - **Disjoint decomposition - subclasses are disjoint**
     - **Exhaustive decomposition - subclasses cover concept**
     - **Partition - exhaustive and disjoint**
3. **Build the Binary relations diagram**
4. **Build the Concept dictionary: Tabulate Concept-Name, Synonym, Instances, Class-Attributes, Instance Attributes, and other Relations**

Methontology: Example

1. **Build the Glossary of terms, define each in natural language**
   carbon, hydrogen, element, gas, metal, conductor…
2. **Build the Concept taxonomy:**

Element
Reactiveness — **Categorisation by reactivity**
Metal
Third transition series



3. **Build the Binary relations diagram**
4. **Build the Concept dictionary**

Concept Name	Synonym	Acronym	Instances	Class Attributes	Instance Attributes	Relations
Element		Elm.			atomic-number atomic-weight	has-structure
Third transition series	Sixth-period transition series	3TS	Gold Hafnium Mercury			

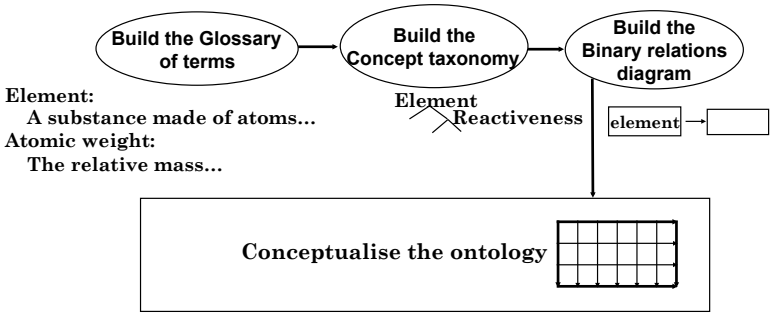
Methontology

Concept dictionary in more detail:

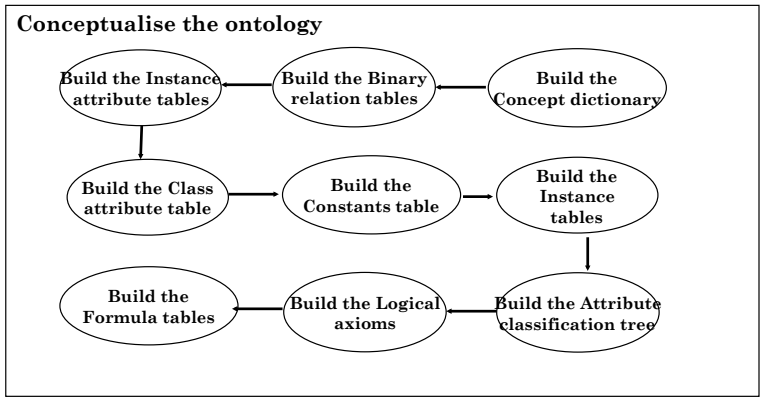
- Concept-Name
  - A unique (singular) name for the concept
  - In certain cases identifiers are used GO:000999
- Synonyms
  - Alternative names for the concept
- Instances
  - Elements of the set represented by the concept
- Relations
  - Relations between (instances of) classes
  - E.g. departurePlace(<instance of Flight> <instance of Location>)
- *[Class-Attributes*
  - *Attributes that hold of the class*
  - *E.g. companyName(KLM-Flight "KLM")*
  - *but arguably class attributes should hold of all instances of the class!]*
- Instance Attributes
  - Attributes that can be specified for each instance
  - E.g. weight(<instance of Person> <weight-in-kg>)

Included here for completeness only

Methontology



Methontology



Methontology: Example 2

A legal ontology for Spanish law [Corcho]

1. Build the Glossary of terms, define each in natural language
   defendant, court, person, private company,…
2. Build the Concept taxonomy:
   LegalEntity
   JuridicialPerson
   Company [exhaustive decomposition]
   PrivateCompany
   PublicCompany
   PhysicalPerson [partitioned into]
   Juvenile
   PersonLegallyOfAge
3. Build the Binary relations diagram
4. Build the Concept dictionary

Juridicial: of law or the administration of justice

Methontology: Example 2

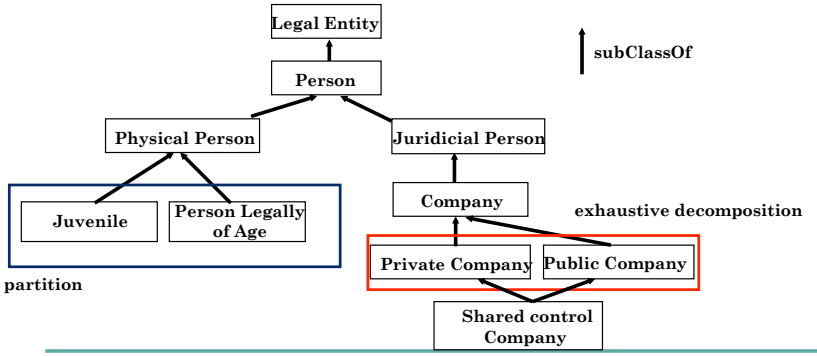
1. Glossary of terms
   - Types are: concept, constant, relation, instance attribute

Name	Synonyms	Acronyms	Description	Type
Adult age in Spain			Adult age is 18	constant
Court	Judicial tribunal		Although court can be understood as a physical place or as a judge, we assume that a court is a judicial tribunal	concept
Birthday			The day when a person was born	instance attribute
Is-defendant (person,lawsuit)			Identifies the lawsuit a defendant is involved in	relation

Judicial: of or by a court or judge

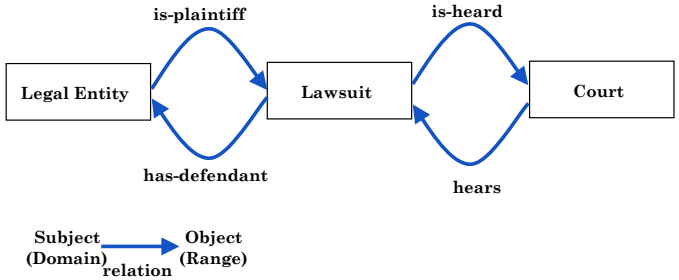
Methontology: Example 2

2. Concept taxonomy: subclasses; partitions; disjoint and exhaustive decompositions



Methontology: Example 2

3. Binary relations diagram
   - Ad-hoc relationships between concepts
   - Determine domains and ranges



Methontology: Example 2

4. Concept dictionary
   - Organise all concepts and relations in 2. and 3.
   - Rows list relations/attributes whose domain is the Concept

Concept Name	Syno-nym	Acro-ynym	Instances	*Class Attrib-utes*	Instance Attributes	Relations
Court			Constitutional Court National Court		number of members seat	hears
Lawsuit						has-defendant has-plaintiff is-heard
Person						is-defendant is-plaintiff
Physical Person					age birthday	is-mother-of has-mother is-father-of

Methontology: Example 2

5. Binary relations table

– Describe all the binary relations in the Concept Dictionary (4.)

Relation Name	Source concept (Domain)	Cardinality	Target concept (Range)	Inverse relation
is-defendant	Person	N	Lawsuit	has-defendant
is-plaintiff	Person	N	Lawsuit	has-plaintiff
has-defendant	Lawsuit	N	Person	is-defendant
is-heard	Lawsuit	N	Court	heard

Methontology: Example 2

6. Instance attributes table

– attributes for instances of concepts (values differ for each instance)

Instance Attribute Name	Domain Concept	Value type	Value range	Cardinality (min,max)
number-of-members	Court	Integer	1..	(1,1)
territorial-jurisdiction	Court	String	---	(1,1)

7. Class attributes table - attributes applicable to the class (values apply to the class name)

Class Attribute Name	Concept	Value type	Cardinality (min,max)	Values
type-of-control	Private Company	[private,public]	(1,2)	private
type-of-control	Public Company	[private,public]	(1,2)	public

Methontology: Example 2

8. Constants table - describe the constants in the Glossary (1.)

Name	Value type	Value	Measurement unit
Adult age in Spain	cardinal	18	year

9. Instances table - describe the instances in the Concept dictionary (4.)

Instance Name	Concept Name	Instance Attribute	Values
National Court	Court	seat territorial-jurisdiction	Madrid Spain

Methontology: Example 2

10. Formula table (Chemical Elements ontology)

– Formula name e.g. Density
– Attribute e.g. density-at-20C
– Description e.g. density is equal to atomic weight divided by atomic volume
– Expression e.g. density-at-20C = atomic-weight / atomic-volume

11. Axioms table

– Axiom name e.g. incompatibility of plaintiff
– Description e.g. a Person cannot be plaintiff and defendant in the same Lawsuit
– Expression …

Recall the axioms for properties of relations

Ontological modelling - Relations

Relations and their properties

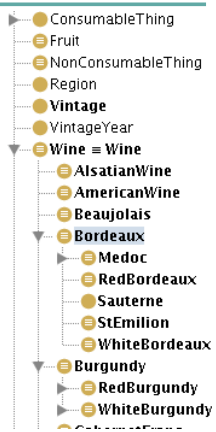
- **Transitive**
  P(x, y) ∧ P(y, z) ⇒ P(x, z)
  part-of(a, b)∧ part-of(b, c) ⇒ part-of(a, c)
- **Symmetric**
  P(x, y) ⇔ P(y, x)
  connected-to(a, b) ⇔ connected-to(b, a)
- **Reflexive**
  P(x, x)
  part-of(a, a)
- **Irreflexive**
  ¬P(x, x)
  ¬hasMother(a, a)
- **Functional**
  P(x, y) ∧ P(x, z) ⇒ y=z
  hasMother
- **Inverse**
  P(x, y) ⇔ Q(y, x)
  hasMother(a, b) ⇔ motherOf(b, a)

Method 1: Methontology

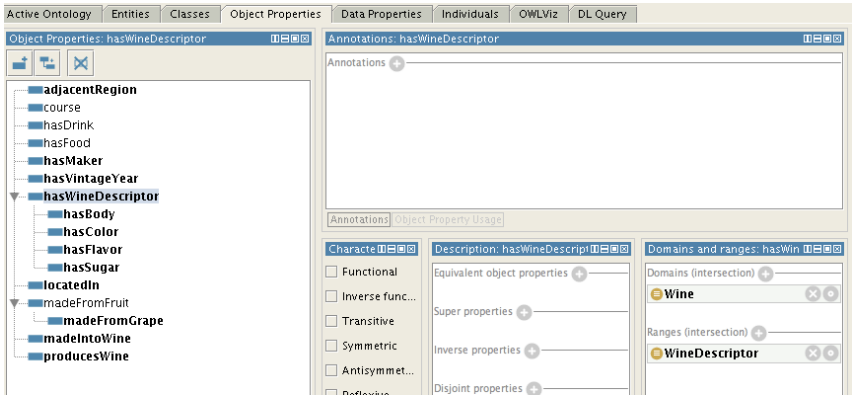
- **A structured methodology based around intermediate representations**
  1. **Glossary of terms**
  2. **Concept taxonomy**
  3. **Binary relations diagram**
  4. **Concept dictionary**
  5. **Binary relations table**
  6. **Instance attributes table**
  7. **Class attributes table**
  8. **Constants table**
  9. **Instances table**
  10. **Formula table**
  11. **Axioms table**

Method 2: Protégé

- **Protégé**
  - **Established, general purpose tool for ontology building**
  - **Install Protégé 4.1 (beta)**
    - » **http://protege.stanford.edu**
    - » **Download installer for linux**
    - » **[dice]:sh install_protege_4.1.bin**
    - » **[dice]: ./run.sh**
    - » **(update the reasoners when prompted)**
  - **Required reading: Ontology Development 101. Noy, N. and McGuinnes, D. Stanford Technical Report (L2-Noy.pdf)**



Method 2: Protégé



Method 2: Protégé

Some development guidelines:

1. Determine the domain and scope of the ontology
   - Competency questions
2. Consider reusing existing ontologies
3. Enumerate important terms in the ontology
4. Define classes and the class hierarchy
   - Bottom-up
   - Top-down
5. Define the properties of classes (slots=attributes or relations)
6. Define facets of slots
   - Domain and range
   - Cardinality
   - Slot value type: string, number, boolean, enumerated, instance (of Classes)

Method 2: Protégé

7. Create instances
8. Ensure the class hierarchy is correct
   - no cycles
   - subClassOf is transitive
9. Classes represent concepts NOT the words that denote the concepts
   - Synonyms are not different classes
   - Shrimps=Prawns
10. Too many siblings may indicate an intermediate class is needed
11. Multiple inheritance is allowed
12. Class or property value?
    White-Wine or Wine & colour=White
13. Instance or class?
14. Singular or plural names for classes?

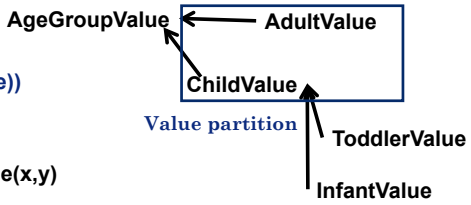
Method 3: Ontology design patterns

Logical partition:

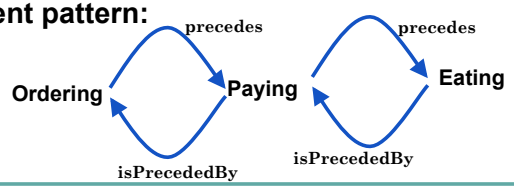
type(y, ChildValue)
=> not(type(y, AdultValue))

Define Adult:

type(x, Adult) ::
type(x, Person) and hasAge(x,y)
and type(y, AdultValue)



Content pattern:



Methodology

From this lecture, note:

- Methontology
  - Specification criteria
  - Conceptualisation process
    - » Use of intermediate representations
    - » Main steps (1-4) in some detail
    - » The remainder in outline
- Protégé
  - Explore use with the example ontologies: family.owl cows.owl and simple-pizza.owl