Semantic Web

Ontologies and the Semantic Web

Motivation

- Ontologies provide a shared understanding of a domain.

- They provide background knowledge to systems to automatize certain tasks.

- By the process of annotation, knowledge can be linked to ontologies.
  - Example: "Angelina Jolie" (Text) linked to concept Actress
  - In our ontology we also know that an actress always is female and a person.

- Ontologies allow the creation of annotations → machine-readable and machine-understandable content.

- If machines can understand content, they can also perform more meaningful and intelligent queries.
  - Distinction of Jaguar the animal and the car.
  - Combination of information that is distributed on the Web.

Motivation

An ontology provides the means for describing explicitly the conceptualization behind the knowledge represented in a knowledge base.

A. Bernaras;I. Laresgoiti; J. Correra.   *Building and Reusing Ontologies for Electrical Network Applications* **ECAI96. 12th European conference on Artificial Intelligence.** Ed. John Wiley & Sons, Ltd. 298-302

- Ontologies are the backbone of the Semantic Web.

- They provide the knowledge that is required for semantic applications of all kinds.

- They are essential to semantic annotation (see previous lecture).

Now we look at a potential application of ontologies in the Ontobroker system for searching and querying the Web. This example illustrates why ontologies are the „backbone of the Semantic Web".

What is an ontology?

- *An ontology defines the basic terms and relations comprising the vocabulary of a topic area, as well as the rules for combining terms and relations to define extensions to the vocabulary*

    Neches, R.; Fikes, R.; Finin, T.; Gruber, T.; Patil, R.; Senator, T.; Swartout, W.R. *Enabling Technology for Knowledge Sharing*. **AI Magazine**. Winter 1991. 36-56

- *An ontology is an explicit specification of a conceptualization*

    Gruber, T. *A translation Approach to portable ontology specifications.* **Knowledge Acquisition**. Vol. 5. 1993. 199-220

What is an ontology (ii)?

- *An ontology is a hierarchically structured set of terms for describing a domain that can be used as a skeletal foundation for a knowledge base*
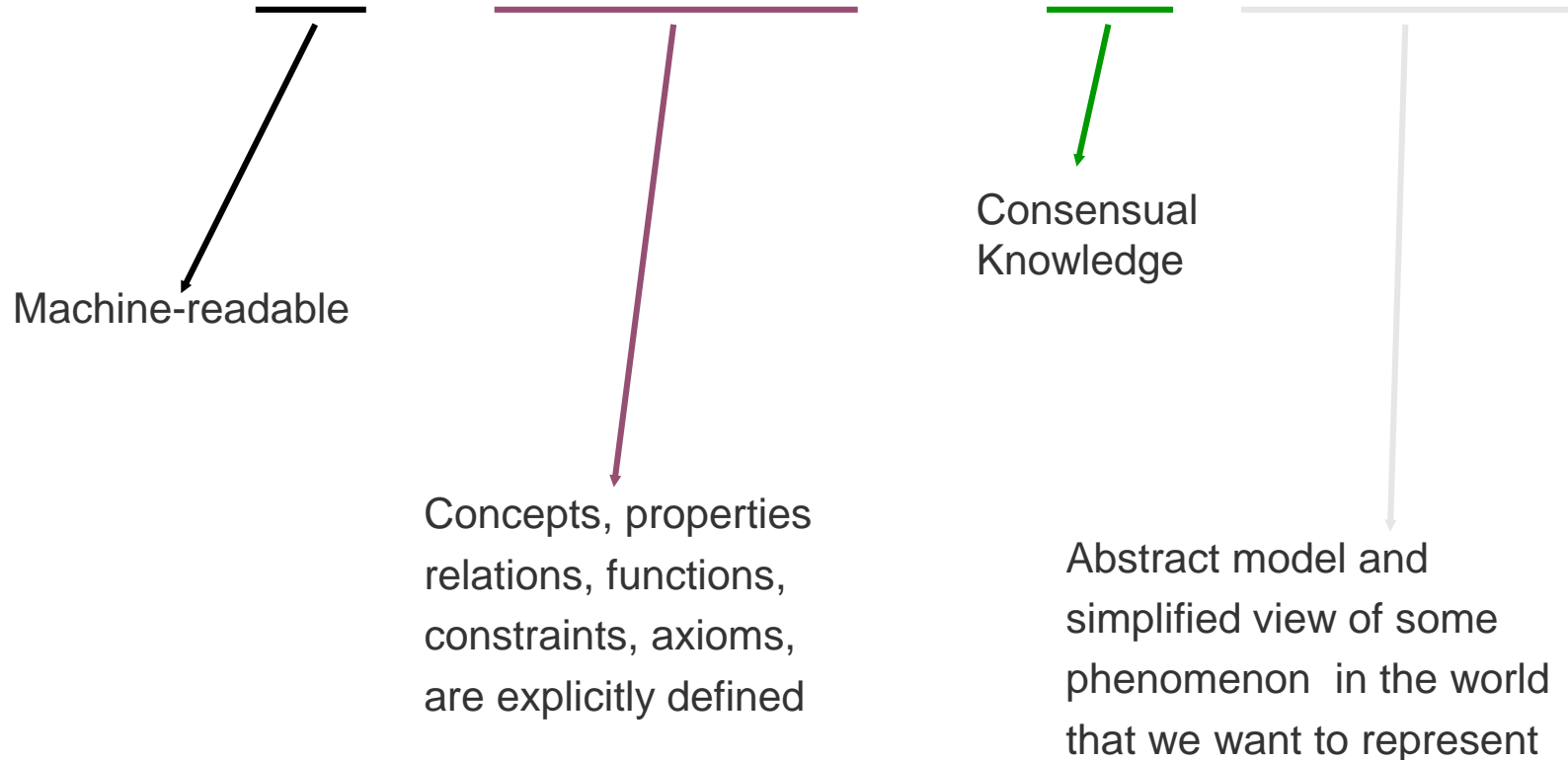
    B. Swartout; R. Patil; k. Knight; T. Russ. *Toward Distributed Use of Large-Scale Ontologies* **Ontological Engineering.** AAAI-97 Spring Symposium Series. 1997. 138-148

- *An ontology provides the means for describing explicitly the conceptualization behind the knowledge represented in a knowledge base*

    A. Bernaras;I. Laresgoiti; J. Correra. *Building and Reusing Ontologies for Electrical Network Applications* **ECAI96. 12th European conference on Artificial Intelligence.** Ed. John Wiley & Sons, Ltd. 298-302
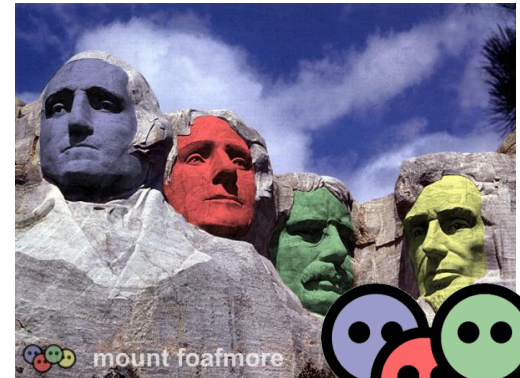
What is an ontology (iii)?

*"An ontology is a formal, explicit specification of a shared conceptualization"*

Machine-readable

Concepts, properties
relations, functions,
constraints, axioms,
are explicitly defined

Consensual
Knowledge

Abstract model and
simplified view of some
phenomenon in the world
that we want to represent

Studer, Benjamins, Fensel. Knowledge Engineering: Principles and Methods. *Data and Knowledge Engineering.* 25 (1998) 161-197

Examples

SUMO

Features of an ontology

- Modelled knowledge about a specific domain

- Defines
    - A common vocabulary
    - The meaning of terms
    - How terms are interrelated

- Consists of
    - Conceptualization and implementation

- Contains
    - Ontological primitives

Classifications of ontologies
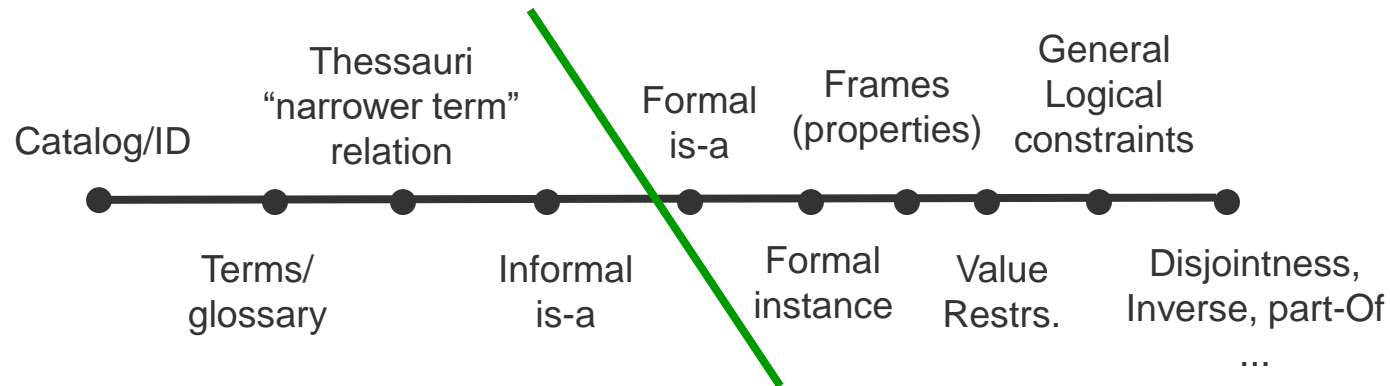
Lassila O, McGuiness D. The Role of Frame-Based Representation on the Semantic Web. Technical Report. Knowledge Systems Laboratory. Stanford University. KSL-01-02. 2001.

Languages for building ontologies

- Ontologies can be built using various languages with various degrees of formality
  - Natural language
  - UML
  - ER
  - OWL/RDFS
  - WSML
  - FOL
  - ...
- The formalism and the language limit the kind of knowledge that can be represented
- A domain model is not necessarily a formal ontology only because it is written in OWL

Applications of ontologies

- Knowledge representation
  - Ontology models domain knowledge

- Semantic annotation
  - Ontology is used as a vocabulary, classification or indexing schema for a collection of items

- Semantic search
  - Ontology is used as a query vocabulary or for query rewriting purposes

- Configuration
  - Ontology defines correct configuration templates

Principles for the design of ontologies

- Clarity:
  - To communicate the intended meaning of defined terms

- Coherence:
  - To sanction inferences that are consistent with definitions

- Extendibility:
  - To anticipate the use of the shared vocabulary

- Minimal Encoding Bias:
  - To be independent of the symbolic level

- Minimal Ontological Commitments:
  - To make as few claims as possible about the world

Gruber, T., *Towards Principles for the Design of Ontologies.* KSL-93-04. Knowledge Systems Laboratory. Stanford University. 1993

Clarity

An ontology should communicate effectively the intended meaning of defined terms. Definitions should be objective. Definitions can be stated on formal axioms, and a complete definition (defined by necessary and sufficient conditions) is preferred over a partial definition (defined by only necessary or sufficient conditions)…

```
(define-class Travel (?travel)
   "A journey from place to place"
:axiom-def
 (and (Superclass-Of Travel Flight)
      (Subclass-Of Travel Thing)
      (Template-Facet-Value Cardinality
           arrivalDate Travel 1)
      (Template-Facet-Value Cardinality
           departureDate Travel 1)
      (Template-Facet-Value Maximum-Cardinality
           singleFare Travel 1))
```

No clarity →

```
:def
  (and (arrivalDate ?travel Date)
       (departureDate ?travel Date)
       (singleFare ?travel Number)
       (companyName ?travel String)))
```

Clarity

```
(define-class Travel (?travel)
   "A journey from place to place"
:axiom-def
 (and (Superclass-Of Travel Flight)
      (Subclass-Of Travel Thing)
      (Template-Facet-Value Cardinality
           arrivalDate Travel 1)
      (Template-Facet-Value Cardinality
           departureDate Travel 1)
      (Template-Facet-Value Maximum-Cardinality
           singleFare Travel 1))
:iff-def
   (and (arrivalDate ?travel Date)
        (departureDate ?travel Date))
:def
   (and (singleFare ?travel Number)
        (companyName ?travel String)))
```

Clarity

Coherence

*An ontology should be coherent: that is, it should sanction inferences
that are consistent with the definitions.[…]
If a sentence that can be inferred from the axioms contradicts a definition
or example given informally, then the ontology is incoherent.*

```
(define-axiom No-Train-between-USA-and-Europe
   "It is not possible to travel by train between the USA and Europe"
:= (forall (?travel)
     (forall (?city1)
      (forall (?city2)
       (=> (and (Travel ?travel)
                (arrivalPlace ?travel ?city1)
                (departurePlace ?travel ?city2)
                (or (and (EuropeanLocation ?city1)
                         (USALocation ?city2))
                    (and (EuropeanLocation ?city2)
                         (USALocation ?city1) )))
            (not (TrainTravel ?travel)))))))
(define-instance Madrid (EuropeanLocation))
(define-instance NewYork (USALocation))
```

Extensibility

> *One should be able to define new terms*
> *for special uses based on the existing vocabulary,*
> *in a way that does not require the revision of the existing definitions.*

- Currency dimension

- Definition of currencies

- Relationship between currencies

```
(define-individual Euro (Unit-of-Measure)
   "An Euro is the currency on the European Union"
:= (* 0,96 USDollar)
:axiom-def
  (= (Quantity.dimension Euro) CurrencyDimension))
```

Minimal Encoding Bias

The conceptualization should be specified at the knowledge level without depending on a particular symbol-level encoding.

```
(define-class Travel (?travel)
   "A journey from place to place"
 :axiom-def
  (and (Superclass-Of Travel Flight)
       (Subclass-Of Travel Thing)
       (Template-Facet-Value Cardinality
            arrivalDate Travel 1)
       (Template-Facet-Value Cardinality
            departureDate Travel 1)
       (Template-Facet-Value Maximum-Cardinality
            singleFare Travel 1))
 :iff-def
   (and (arrivalDate ?travel Date)
        (departureDate ?travel Date))
 :def
   (and (singleFare ?travel Number)
        (companyName ?travel String)))
```

No minimal encoding bias

Minimal Encoding Bias

```
(singleFare ?travel Number)

should be substituted by:

(singleFare ?travel CurrencyQuantity)
```

Physical-Quantities Ontology

Standard-Dimensions Ontology

Instance-of — Ampere
Amu
Angstrom
.
.
.
Volt
Watt
Year

Unit-of-Measure

Instance-of

Euro

....
Density-Dimension
....
Frequency-Dimension
....
Length-Dimension
Mass-Dimension
....
Pressure-Dimension
Resistance-Dimension
......
Work-Dimension

Subclass-of

Instance-of

Instance-of

System-of-Units

Si-Unit

Instance-of — Ampere
Candela
Degree-Kelvin
Identity-Unit
Kilogram
Meter
Mole
Second-of-Time

Instance-of

Physical-Dimension

Instance-of

Currency Dimension

Minimal Ontological Commitments

Since ontological commitment is based on the consistent use of the vocabulary, ontological commitment can be minimized by specifying the weakest theory and defining only those terms that are essential to the communication of knowledge consistent with the theory.

```
(define-class Travel (?travel)
   "A journey from place to place"
:axiom-def
 ( .... )
:iff-def
  (and (arrivalDate ?travel Date)
       (departureDate ?travel Date))
:def
  (and (singleFare ?travel Number)
       (companyName ?travel String)))
```

- Absolute/relative date?
- Format?
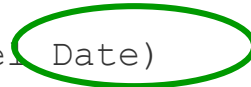
Ontological Commitments

- Agreements to use the vocabulary in a coherent and consistent manner (Gruber)
  - An agent commits (conforms) to an ontology if it "acts" consistently with the definitions

- Connection between the ontology vocabulary and the meaning of the terms of such vocabulary

Ontological commitments

WordNet — a lexical database for the English language

cognitive science laboratory | princeton university | 221 nassau st. | princeton, nj 08542

Search word: flight | Find valid searches

WordNet 1.6 overview for "flight"

The **noun** "flight" has 9 senses in WordNet.

1. **flight** – (a formation of aircraft in flight)
2. **flight**, flying – (an instance of traveling by air; "flying was still an exciting adventure for him")
3. **flight**, flight of stairs, flight of steps – (a set of steps between one floor or landing and the next)
4. escape, **flight** – (the act of escaping physically; "he made his escape from the mental hospital", "the canary escaped from its cage", "his flight was an indication of his guilt")
5. **flight** – (a unit of the US air force smaller than a squadron)
6. **flight** – (passing above and beyond ordinary bounds; "a flight of fancy", "flights or rhetoric", "flights of imagination")
7. trajectory, **flight** – (the path followed by a moving object)
8. **flight** – (a flock of flying birds)
9. **flight** – (a scheduled trip by plane between designated airports; "I took the noon flight to Chicago")

Principles for the design of ontologies

- **The representation of disjoint and exhaustive knowledge**
  - If the set of subclasses of a concept are disjoint, we can define a disjoint decomposition. The decomposition is exhaustive if it defines the super-concept completely.

- **The standardization of names.**
  - To ease the understanding of the ontology the same naming conventions should be used to name related terms.
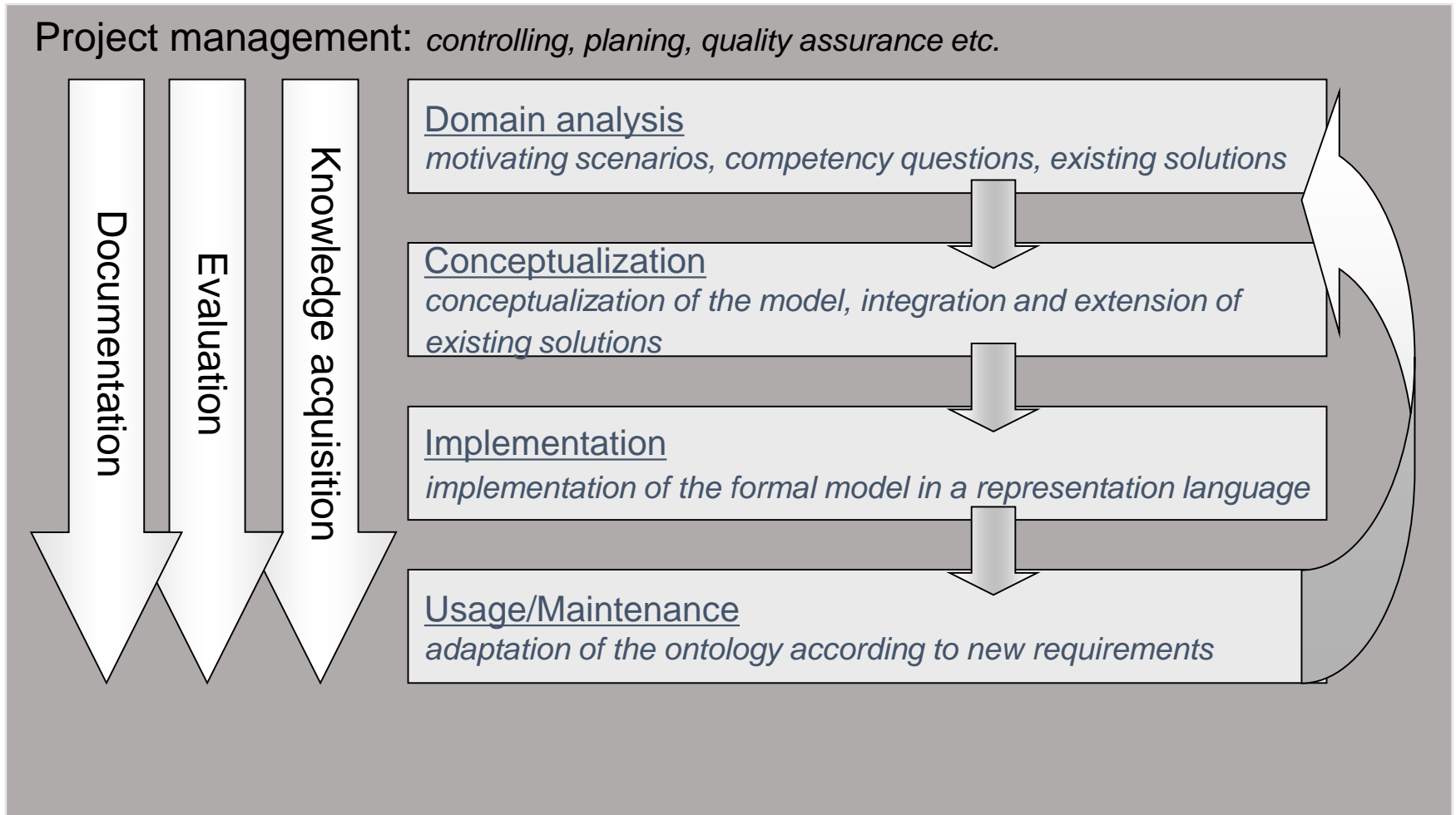
Arpírez JC, Gómez-Pérez A, Lozano A, Pinto HS (1998) *(ONTO)²Agent: An ontology-based WWW broker to select ontologies*. In: Gómez-Pérez A, Benjamins RV (eds) ECAI'98 Workshop on Applications of Ontologies and Problem-Solving Methods. Brighton, United Kingdom, pp 16–24

ONTOLOGY ENGINEERING

What is ontology engineering?

- *"the set of activities that concern the ontology development process, the ontology life cycle, and the methodologies, tools and languages for building ontologies"*
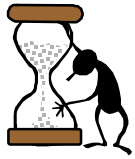
<div align="right">Gomez-Perez et al, 2004</div>
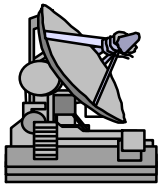
Ontology life cycle

Project management: *controlling, planing, quality assurance etc.*

Documentation

Evaluation

Knowledge acquisition

Domain analysis
motivating scenarios, competency questions, existing solutions

Conceptualization
conceptualization of the model, integration and extension of existing solutions

Implementation
implementation of the formal model in a representation language

Usage/Maintenance
adaptation of the ontology according to new requirements

Ontology engineering activities

Management	Development oriented	Support

Management

Scheduling

Control

Quality assurance

Development oriented

Pre-development
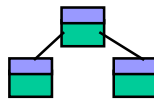
Environment study
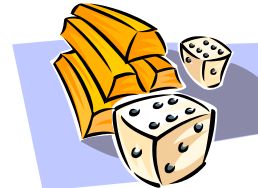
Feasibility study

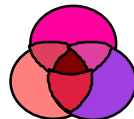*Development*

Specification

Conceptualization

Formalization

Implementation

Post-development

Maintenance

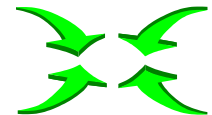Use

Support

Knowledge acquisition
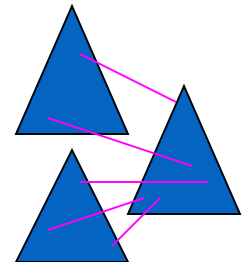
Evaluation

Integration

Documentation

Merging

Configuration management

Alignment

Methodologies for building ontologies

- Validated guidelines on how the ontology building process should be structured

- Well-organized process instead of „ontology development driven by inspiration and intuition only"

- Methodologies do not support all of the aforementioned activities

- They implicitly assume a particular development paradigm for the ontology engineering process

- Some of them also provide supporting methods and tools

Collaborative ontology engineering

- Two or more people interact and exchange knowledge in order to build a common, shared ontology in pursuit of a shared, collective, bounded goal.
  - *Interaction may be indirect but required.*
  - *Argumentation as a common interaction means.*
  - *Simple contributions not enough.*
  - *Bounded goal: beginning and end.*
  - *Collaborators may have individual goals.*

Uschold and King Methodology

- Identify purpose

- Build ontology
  - Capture
  - Coding
  - Integrating

- Evaluation

- Documentation

Grüninger and Fox Methodology

- Identify motivating scenarios
- Elaborate informal competency questions
- Specify terminology in FOL
    - Identify objects
    - Identify predicates
- Formal competency questions
- Specify axioms in FOL
- Specify completeness theorems

METHONTOLOGY methodology

- Develop + Management Activities + Support in parallel

- Develop:
    - Specification
    - Conceptualization
    - Formalization
    - Implementation
    - Maintenance

- Focus on the conceputalization activity
    - Advantage: Integration of existing ontologies considered from early on
    - Conceptualization is evaluated early on, which prevents propagation of errors.

.

On-To-Knowledge Methodology

- Feasibility study
- Kick-off: describe domain and goals, design guidelines (e.g. naming conventions), available sources
- Refinement
  - Knowledge elicitation with domain experts
 - Formalization
- Evaluation
- Maintenance

DILIGENT

1. **Build**: a small team builds an ontology which is not required to be complete.

2. **Local adaption**: users adapt it to their own needs in their local environments.

3. **Analysis**: the board selects which changes go to the next version of the ontology.

4. **Revision**: carried out regularly to avoid that the ontology diverges too far from the local adaptions.

5. **Local** updates may be performed if the users wish to align their local ontologies with the new version of the ontology.
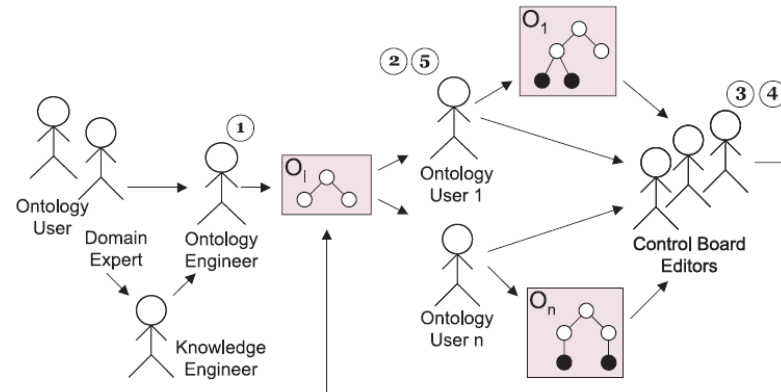
Vrandecic et al., 2005

Ontology Maturing

1.   Emergence of ideas. In the first phase, new concept ideas are collected in an ad-hoc fashion. This is done using simple tags.

2.   Consolidation in communities. The concept symbols generated in the first phase are re-used and adapted by the user community. The phase aims at extracting concepts from the available tags leading to a common terminology.

3.   Formalization. This phase adds taxonomic and ad-hoc relations to the common terminology yielding lightweight but formal ontologies.

4.   Axiomatization. The final step in the methodology addresses knowledge workers, i.e. ontology engineers, to add axiom leading to a heavy-weight ontology.
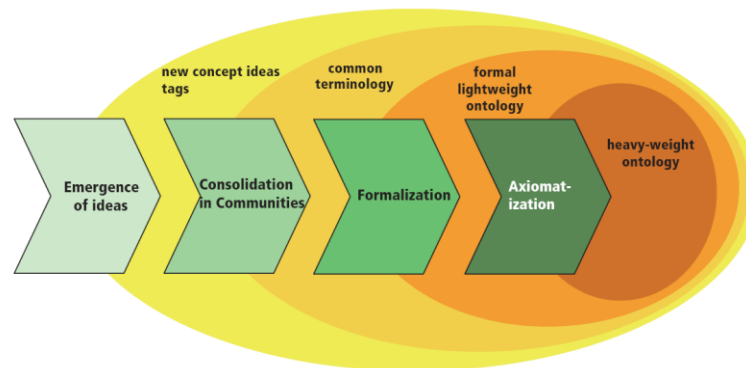
Holsapple and Joshi's approach

1.   Preparation phase defines design criteria, determines boundary conditions and determines standards that can then be used for evaluation.

2.   Anchoring produces a first ontology that helps for orientation of the participants. This ontology is refined in several steps.

3.   Iterative improvement is an adaptation of the Delphi method, a technique for collecting views of several stakeholders. This process is repeated until consensus is reached.

4.   Application is the actual usage of the ontology.

UPON

1. Requirements workflow
   a) Determining the domain of interest and scope.
   b) Defining the purpose involves a description of the application of the ontology or a motivating scenario.
   c) Define storyboard and terminology.
   d) Creating an application lexicon can be carried out in a semi-automatic fashion that can be supported by automatic tools that extract knowledge from documents.
   e) Identifying the competency questions.
   f) Use case identification.
2. The analysis workflow depicts the conceptual analysis of the domain in order to derive ontology requirements and ontological commitments.
   a) Considering reuse of existing resources.
   b) Modeling the application scenario using UML diagrams.
   c) Building the glossary.
3. The design workflow refines results of previous steps and specifies relationships.
4. The implementation workflow leads to a serialized ontology in the preferred ontology representation formalism. The authors recommend OWL.
5. The test workflow validates the ontology by testing against the motivation scenario.

HCOME

1. Specification.
   a) identified the collaborators (team building), the knowledge workers
   b) discuss requirements, produce specification documents, and
   c) reach consensus on the scope and aim of the ontology.

2. Conceptualization first takes place in personal spaces addressing the following tasks:
   a) import of existing ontologies from ontology libraries,
   b) consult generic top ontologies for better understanding,
   c) improvising of ontologies, i.e. from-scratch-development, based on domain experts' views,
   d) management, mapping and merging of various ontology versions,
   e) comparison of available ontology versions,
   f) enriching ontology concepts with more information and specification details.

3. Exploitation. In this phase, the developed ontologies are pushed from the personal spaces to the shared space in order to reach a common understanding via structured conversation and criticism.
   a) inspection of agreed or shared ontologies by collaborators,
   b) comparisons of versions of one ontology in order to spot differences, and
   c) publication of comments and feedback.

HOW TO BUILD AN ONTOLOGY

Natalya F. Noy and Deborah L. McGuinness. ``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, March 2001.

Step 1: Determine the domain and scope of the ontology

- What is the domain that the ontology will cover?

- For what we are going to use the ontology?

- For what types of questions the information in the ontology should provide answers?

- Who will use and maintain the ontology?

Competency Questions

- A set of queries which place demands on the underlying ontology.

- Ontology must be able to represent the questions using its terminology and the answers based on the axioms

- Ideally, in a staged manner, where consequent questions require the input from the preceeding ones.

- A rationale for each competency question should be given.

Step 2: Consider reusing existing ontologies

- Reuse ensures interoperability and reduces costs

- Ontology libraries and tools for customization are required for this step

- Sub-steps
  - Discover potential reuse candidates
  - Evaluate their usability
  - Customize ontologies to be reused
  - Integrate and merge to the target ontology

Step 3: Enumerate important terms in the ontology

- What are the terms we would like to talk about?

- What properties do those terms have?

- What would we like to say about those terms?

Step 4: Define classes and class hierarchy

- A top-down development process starts with the definition of the most general concepts in the domain and subsequent specialization of the concepts.

- A bottom-up development process starts with the definition of the most specific classes, the leaves of the hierarchy, with subsequent grouping of these classes into more general concepts.

- Middle-out approach: define the more salient concepts first and then generalize and specialize them appropriately.

Step 4: Define classes and class hierarchy (ii)

- From the list created in Step 3, select the terms that describe objects having independent existence rather than terms that describe these objects.
  - These terms will be classes in the ontology.
- Organize the classes into a hierarchical taxonomy by asking if by being an instance of one class, the object will necessarily (i.e., by definition) be an instance of some other class.
  - *If a class A is a superclass of class B, then every instance of B is also an instance of A.*
- Classes as unary predicates—questions that have one argument. For example, "Is this object a wine?"
  - Later: binary predicates (or slots)—questions that have two arguments. For example, "Is the flavor of this object strong?" "What is the flavor of this object?"

How to find classes

- Interview: talk to subject matter experts.

- Documentation: read what experts have written about the subject matter, read the requirements documentation, read proposals and invitations to tender.

- Observation and reflection.

- Typical candidates for classes: NOUNS.
  - But: actors of use cases do not necessarily correspond to classes.

- Other terms as well
  - Gerund: „My eyes glazing over…" ~ withdrawal
  - Verbs: an association which starts to take on attributes and associations of its own turns into an entity: „Officer arrests suspect".
  - Verbs: events: „Illness episode".
  - Passive form: re-formulate in active form.
  - No pronouns.

Cohesion and identity

- A class should represent one thing, all of that thing and nothing but that thing.

- You can prove cohesion by
  - Giving the class a representative name.
  - Noun (+ adjective, sometimes however also captured as attribute value).
    - Blackmail victim, robbery victim.
    - Blue car, red car.
    - **Cars** is not cohesive.

- Avoid ambiguous terms.
  - Manager, handler, processor, list, information, item, data…

- Instances vs classes of things.

- Identity ~ individuality: entities change values, but are still the same entity
  - Child/Adult: age

Relevance

- Goint out too far vs. going down too far.

- Investigate homonyms and synonims.
  - Can medicine and drug be considered synonims?
  - Do they have the same properties/characteristics/attributes/relationships?
  - Do they have a critical mass of commonalities?

Class hierarchy

- A subclass of a class represents a concept that is a "kind of" the concept that the superclass represents.

- Classes represent concepts in the domain and not the words that denote these concepts. Synonyms for the same concept do not represent different classes.

- All the siblings in the hierarchy (except for the ones at the root) must be at the same level of generality.

- If a class has only one direct subclass there may be a modeling problem or the ontology is not complete.

- If there are more than a dozen subclasses for a given class then additional intermediate categories may be necessary.

- Subclasses of a class usually (1) have additional properties that the superclass does not have, or (2) restrictions different from those of the superclass, or (3) participate in different relationships than the superclasses.

- If the concepts with different slot values become restrictions for different slots in other classes, then we should create a new class for the distinction. Otherwise, we represent the distinction in a slot value.

Examples

Bill	MealOrder
Dish	Menu
Bed	Mattress
Diary	Appointment
Crime	CrimeScene

Step 5: Define attributes and relationships

- Step 4 selected classes from the list of terms we created in Step 3.
  - Most of the remaining terms are likely to be properties of these classes.
  - For each property in the list, we must determine which class it describes.
- Types of properties
  - "intrinsic" properties
  - "extrinsic" properties
  - parts, if the object is structured (physical or abstract).
  - relationships to other individuals.
- Properties are inherited and should be attached to the most general class in the hierarchy.

Characterizing classes

- Two types of principal characteristics:
  - Measurable properties: attributes.
  - Inter-class connections: relationships.

  - Use relationships to capture something with an identity.

  - Arrest details as attribute of the suspect vs. arrest as an relationship.
    - Do we measure degrees of arrestedness or do we want to be able to distinguish between arrests?

  - Color of an image as attribute vs. class.

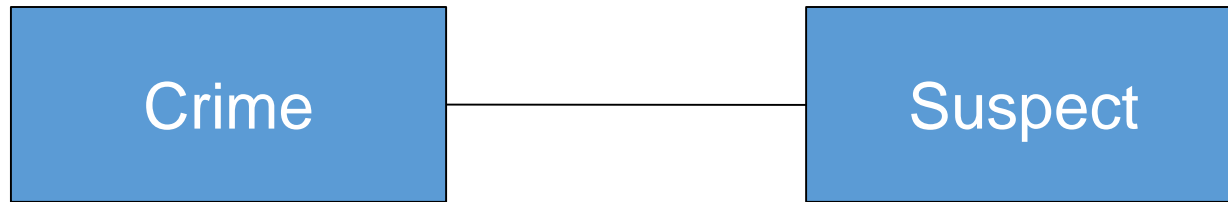 - A „pointing finger" rather than a „ruler" indicates identity.

How to find attributes

- Interview: talk to subject matter experts.
- Documentation: read what experts have written about the subject matter, read the requirements documentation, read proposals and invitations to tender.
- Observation and reflection.

- Nouns in „-ness"
  - Velocity-ness, job-ness, arrested-ness…

- „How much, how many" test.
  - If you evaluate this, then it is probably an attribute.
  - If you enumerate these, it is probably an entity.

Relationships

- Are defined on sets of instances.

- Properties: reflexivity, cardinality, functional, inverse-functional, discountinuous multiplicity, many-to-many, all values from, some values of, transitivity, symmetry etc.

- Arity.

Examples
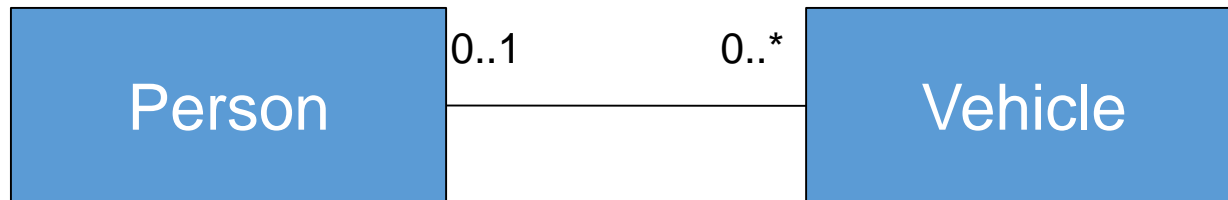
How to find relationships

- Interview: talk to subject matter experts.
- Documentation: read what experts have written about the subject matter, read the requirements documentation, read proposals and invitations to tender.
- Observation and reflection.

- Verbs, verbal phrases and things that could have been verbs.
 - *„The buttler murdered the duchess"*

Roles

- *Never draw more than three relationships between classes without going back and starting to label them.*

- The most common way to label relationships is with role names.

- Nouns, adjectives.

- Verbs: indication of time's passing.
  - Short-term, one-to-one associations should be named with present participles.
  - Longer-term, one-to-many associations should be named with past participles, or with the simple present third-person singular.

Examples
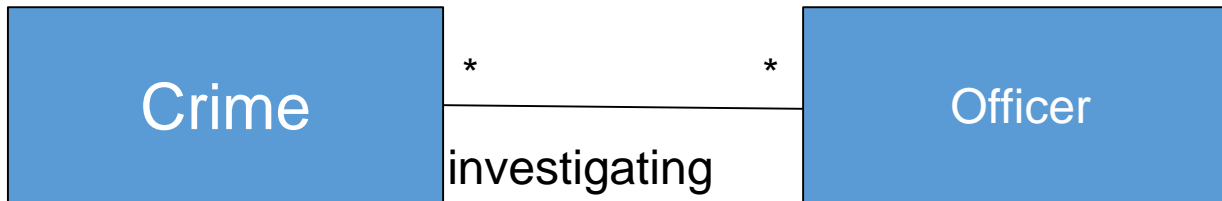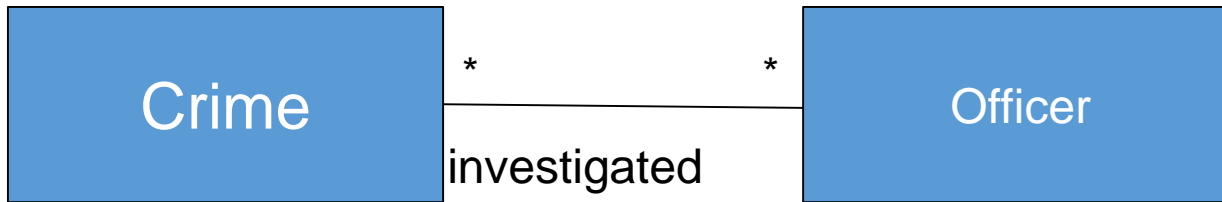
Part-of

- Describes the relationship between wholes and parts.

- Existence dependence: Component is mandatory (or not) for the existence of the composit.
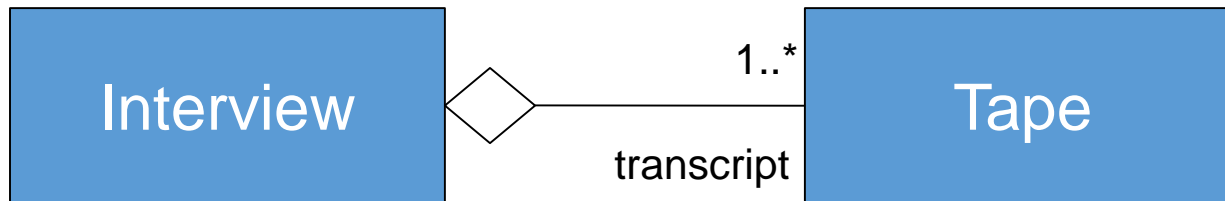
- 1..* vs existence-dependence.

Step 6: Define the restrictions of the properties

- Refine the semantics of the properties
  - Cardinality
  - Domain and range
    - When defining a domain or a range for a slot, find the most general classes or class that can be respectively the domain or the range for the slots .
    - Do not define a domain and range that is overly general

Step 7: Create instances

- Define an individual instance of a class requires
  - choose a class
  - create an individual instance of that class
  - filling in the values of the properties

Questions?