Ontologies for the Intelligence Community (OIC) 2009 Tutorial:

Information Semantics 101: Semantics, Semantic Models, Ontologies, Knowledge Representation, and the Semantic Web

Dr. Leo Obrst
Information Semantics
Information Discovery & Understanding
Command & Control Center
MITRE
Lobrst@mitre.org
October 20, 2009

Copyright © Leo Obrst, MITRE, 2009
Overview

• This course introduces Information Semantics, i.e., Semantics, Semantic Models, Ontologies, Knowledge Representation, and the Semantic Web
• Presents the technologies, tools, methods of ontologies
• Describes the Semantic Web and emerging standards

Brief Definitions (which we’ll revisit):

• **Information Semantics**: Providing semantic representation for our systems, our data, our documents, our agents
• **Semantics**: Meaning and the study of meaning
• **Semantic Models**: The *Ontology Spectrum*: Taxonomy, Thesaurus, Conceptual Model, Logical Theory, the range of models in increasing order of semantic expressiveness
• **Ontology**: An ontology defines the terms used to describe and represent an area of knowledge (subject matter)
• **Knowledge Representation**: A sub-discipline of AI addressing how to represent human knowledge (conceptions of the world) and what to represent, so that the knowledge is usable by machines
• **Semantic Web**: "The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation."

Schedule

• **Morning**
  – 9:00-10:20: Part 1: Syntax, Semantics, Ontology Spectrum, Taxonomies
  – 10:20-10:40: Break
  – 10:40-12:00: Part 2: Thesauri, Conceptual Models, Logical Theories (Strong Ontologies)
  – 12:00-1:20: Lunch

• **Afternoon**
  – 2:40-3:00 Break
  – 3:00-4:20: Part 4: The Semantic Web
Agenda, Part 1:
Semantics, Semantic Models, and Ontologies
The Problem

• With the increasing complexity of our systems and our IT needs, we need to go to human level interaction
• We need to maximize the amount of Semantics we can utilize
• From data and information level, we need to go to human semantic level interaction

And represented semantics means multiply represented semantics, requiring semantic integration
The Solution

- We need to offload the very real, heavy cognitive interpretation burden from humans to our systems.

- We need to represent human semantics using machine-interpretable ontologies.
Advancing Along the Interpretation Continuum

Simple Metadata: XML

Human interpreted

DATA
• Relatively unstructured
• Random

Interpretation Continuum

Computer interpreted

KNOWLEDGE
• Very structured
• Logical

Richer Metadata: RDF/S

Very Rich Metadata: DAML+OIL

Moving to the right depends on increasing automated semantic interpretation

• Info retrieval
• Web search
• Text summarization
• Content extraction
• Topic maps
• Reasoning services
• Ontology Induction

Display raw documents; All interpretation done by humans
Find and correlate patterns in raw docs; display matches only
Store and connect patterns via conceptual model (i.e., an ontology); link to docs to aid retrieval
Automatically acquire concepts; evolve ontologies into domain theories; link to institution repositories (e.g., MII)
Automatically span domain theories and institution repositories; inter-operate with fully interpreting computer

Copyright © Leo Obrst, MITRE, 2002-09
Motivation: Tightness of Coupling & Semantic Explicitness

Implicit, TIGHT

Explicit, Loose

Looseness of Coupling

Performance = k / Integration_Flexibility

Data

Application

1 System: Small Set of Developers

Copyright © Leo Obrst, MITRE, 2002-09
Information Semantics

- Provide semantic representation (meaning) for our systems, our data, our documents, our agents
- Focus on machines more closely interacting at human conceptual level
- Spans Ontologies, Knowledge Representation, Semantic Web, Semantics in NLP, Knowledge Management
- Linking notion is Ontologies (rich formal models)
- **Content is King or should be!**
  - And the better the content…
It All Depends on What ‘is’ is

• Semantics is meaning
• “Oh, it’s just semantics”: Wrong!
  – Implies that it’s quibbling about meaning, i.e., meaningless meaning, mincing words, not substantive or contentful distinctions
• “Real” semantics is about meaning
  – What meaning do we assign our squiggles on the page, pixels on the screen, ink on a map, sounds in a track, bits on a disk, flickering shades of dark & light on a film, squinting of an eye, a shrug?
  – What is the meaning of: ‘45-XG-92+@55’?
  – Is it the same or similar to ‘abk3#40’?
  – What is the meaning of ‘the man hit the ball’? ‘Green ideas sleep furiously’? ‘Hit man the the ball’? ‘Joe is a abk3#40’?
  – It’s the meaning of systems, data, document, agents, humans
Semantics

• Semantics is meaning
  – Literal & figurative
  – Both context independent & context dependent
  – Meaning & use (intent of the meaning)
  – Natural language, programming & formal languages
  – Informal & formal
  – Express the meaning in a loose/strict, natural language definition or description
    • Semantics (Merriam-Webster, http://www.m-w.com/cgi-bin/dictionary)
      1 : the study of meaning: a : the historical and psychological study and the classification of changes in the signification of words or forms viewed as factors in linguistic development b (1) : semiotic (2) : a branch of semiotics dealing with the relations between signs and what they refer to and including theories of denotation, extension, naming, and truth.
  – Express the meaning in a logical, mathematically rigorous manner
    • All students who took the test passed.
      \( \forall x: (\text{student}(x) \land \text{took_test}(x) \rightarrow \text{passed_test}(x)) \)

• Syntax vs. Semantics: based on Language
Syntax

- A Language has a Syntax (set of symbols, & formation rules) & a Semantics (what the symbols, well-formed formulas mean)
- A formal language can be identified by its set of well-formed formulas; a natural language by its set of sentences (infinite)
- Syntax is form & structure
  - Symbols
  - Tokens/Types
    - Restricted words of a programming language
      - Do, While, Until, If, Then, Else, Declare
    - User defined constants & variables
      - A = 7 + 3; Y = A + 1; While Count < 5 Do
  - Order: how do words combine
    - To form a program?
    - To form a sentence?
    - Rules for combining
- Applies to Natural Languages, Programming Languages, Formal Languages, including Logics, Knowledge Representation/Ontology Languages!
Syntax: Propositional Logic

- PL is a Language having a Syntax & a Semantics
  - A set of symbols:
    - Logical Constants: True, False (or T, F)
    - Logical Variables (or propositional symbols): p, q, r, ...
    - Logical Operators (or connectives): \( \neg, \lor, \land, \rightarrow, \leftrightarrow, (, ) \)
  - Formulas (Well-formed Formulas or WFFs) of PL (we will call these *propositions*)
    - Any propositional symbol is a WFF of PL
    - If \( \alpha \) and \( \beta \) are WFFs, then so are \( (\alpha \lor \beta), (\alpha \land \beta), (\alpha \rightarrow \beta), (\alpha \leftrightarrow \beta) \), and \( (\neg \alpha) \) [and note that we will dispense with parentheses where we can]
    - Nothing else is a WFF.
  - So the following are WFFs: \( p, \neg p, p \lor q, p \land q, (p \land q) \rightarrow r \)
  - Propositions are things that are true or false

**Propositions in English:**
If John is a management employee, then John manages an organization.

John is a management employee.

John manages an organization (MP)

**Propositions in PL:**
\[ p \rightarrow q \]
\[ p \]
\[ q \quad \text{(MP: Modus Ponens)} \]

Still Need Semantics!
Predicate Logic: Add Predicates, Individuals, Quantifiers

Propositions & Predicates in English:
If John is a management employee, then John manages an organization.
John is a management employee.
John manages an organization (MP)

Propositions & Predicates in First Order Predicate Logic:
\[ \forall x. [p(x) \rightarrow \exists y. [q(y) \land r(x,y)]] \]
"For all x, if x is a p, then there is some y such that y is a q, and x is in the r relation to y”
p(john)
\[ \exists y. [q(y) \land r(john,y)] \]
(MP: Modus Ponens)

Still Need Semantics!
Semantics: Interpretation

• Interpretation:
  – An *interpretation* of a formal language is an assignment of meanings to its symbols and/or formulas [Hunter, 1973, p.6-7]
  – “An interpretation of PL is an assignment to each propositional symbol (logical variable) of one or other (but not both) of the truth values truth and falsity” [Hunter, 1973, p. 57-58, over next few slides]

• Truth tables:
  \[ \neg p \lor (q \land r) \equiv (p \rightarrow q) \land (p \rightarrow r) \]

<table>
<thead>
<tr>
<th>p</th>
<th>q</th>
<th>r</th>
<th>\neg p</th>
<th>(q \land r)</th>
<th>\neg p \lor (q \land r)</th>
<th>(p \rightarrow q)</th>
<th>(p \rightarrow r)</th>
<th>(p \rightarrow q) \land (p \rightarrow r)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Proof Theory (Syntax) vs. Model Theory (Semantics)

- **Proof Theory**: deductive apparatus of a language
  - **Axioms**: declaring by fiat certain formulas of L
  - **Rules of Inference**: determines which relations between formulas of L are relations of *immediate consequence* of L
    - i.e., from $\alpha \implies \beta$ in one step
    - More generally, **syntactic consequence** is: iff there is a derivation in PL of the set of formulas $\beta$ from the set of formulas $\alpha$, written $\alpha \vdash \beta$
  - Apply rules to Axioms to derive Theorems
  - **Theorem**: a formula of a formal language that satisfies purely syntactic requirements and has no meaning

- **Formal Model**: a model of a formula of L is an interpretation of L for which the formula comes out true (a proposition)

- **Model Theory**: the theory of interpretations of languages
  - **Logical Validity**: `$\models \alpha$’ means that $\alpha$ is a logically valid formula of PL iff $\alpha$ is true for every interpretation of PL
  - **Semantic consequence**: `$\alpha \models \beta$’ means $\beta$ is a semantic consequence of $\alpha$ iff there is no interpretation of PL for which $\alpha$ is true and $\beta$ is false
There is no single real elephant

There must be an upper elephant

An elephant is abstract

An elephant is very abstract

There must be a purpose for an elephant: use cases?

An elephant is really very simple

There are only distributed elephants & their mappings

An elephant is the result of consensus

Open vs. Closed Elephant

Copyright © Leo Obrst, MITRE, 2002-09
Some Issues

• We are like the blind men & the elephant: describing the ontology elephant from our own perspectives, which is of course what we most know about

• Multiple communities converging on semantics, with their own perspectives, concepts: see Ontology Spectrum
  – Logicians, formal ontologists, formal semanticists, some computer scientists
  – Librarian, information scientists
  – Object-oriented, development, programmers & software engineers
  – Classical AI knowledge representation folks
  – Database theorists & practitioners
  – Web community
  – Service Oriented Architecture (SOAs), Web services, enterprise architecture folks
  – Business & government analysts

• Problems:
  – Key distinctions are glossed over: term vs. concept, label vs. model, machine vs. human interpretability, syntax vs. semantics-pragmatics (sense, reference, discourse, speech acts)
Ontology & Ontologies 1

• An ontology defines the terms used to describe and represent an area of knowledge (subject matter)
  – An ontology also is the model (set of concepts) for the meaning of those terms
  – An ontology thus defines the vocabulary and the meaning of that vocabulary

• Ontologies are used by people, databases, and applications that need to share domain information
  – Domain: a specific subject area or area of knowledge, like medicine, tool manufacturing, real estate, automobile repair, financial management, etc.

• Ontologies include computer-usable definitions of basic concepts in the domain and the relationships among them
  – They encode domain knowledge (modular)
  – Knowledge that spans domains (composable)
  – Make knowledge available (reusable)
The term *ontology* has been used to describe models with different degrees of structure (Ontology Spectrum)

- **Less structure:** Taxonomies (Semio/Convera taxonomies, Yahoo hierarchy, biological taxonomy, UNSPSC), Database Schemas (many) and metadata schemes (ICML, ebXML, WSDL)
- **More Structure:** Thesauri (WordNet, CALL, DTIC), Conceptual Models (OO models, UML)
- **Most Structure:** Logical Theories (Ontolingua, TOVE, CYC, Semantic Web)

Ontologies are usually expressed in a logic-based language

- Enabling detailed, sound, meaningful distinctions to be made among the classes, properties, & relations
- More expressive meaning but maintain “computability”

Using ontologies, tomorrow's applications can be "intelligent"

- Work at the human conceptual level

Ontologies are usually developed using special tools that can model rich semantics
Big O: Ontology, Little o: ontology

- **Philosophy:** “a particular system of categories accounting for a certain vision of the world” or domain of discourse, a conceptualization (Big O)

- **Computer Science:** “an engineering product consisting of a specific vocabulary used to describe a part of reality, plus a set of explicit assumptions regarding the intended meaning of the vocabulary words”, “a specification of a conceptualization” (Little o)

- **Ontology Engineering:** towards a formal, logical theory, usually ‘concepts’ (i.e., the entities, usually classes hierarchically structured in a special subsumption relation), ‘relations’, ‘properties’, ‘values’, ‘constraints’, ‘rules’, ‘instances’, so:

- **Ontology (in our usage):**
  1) A logical theory
  2) About the world or some portion of the world
  3) Represented in a form semantically interpretable by computer
  4) Thus enabling automated reasoning comparable to a human’s

* The first two definitions are derived from Guarino, 98; Guarino & Giaretta, 95; Gruber, 93, 94
Ontology thus includes:

• **Objects** (things) in the many domains of interest
• The **relationships** between those things
• The **properties** (and property values) of those things
• The **functions and processes** involving those things
• **Constraints** on and **rules** about those things
 Ontology Spectrum: Generality & Expressiveness

- **Conceptual Model**: Sub-Classification of Model, XML
  - Problem: Local
  - Semantic Expressivity: Low

- **Thesaurus**: Narrower Meaning Than, DB Schemas, XML Schema
  - Problem: General
  - Semantic Expressivity: Medium

- **Taxonomy**: Sub-Classification of Model, XML
  - Problem: Local
  - Semantic Expressivity: Low

- **Description Logic**, **DAML+OIL, OWL**, **UML**, **First Order Logic**, **Modal Logic**, **RDF/S**, **XTM**
  - Problem: Very General
  - Semantic Expressivity: Very High

- **Logics**:
  - **Strong semantics**
  - **Weak semantics**

- **Features**:
  - **Semantic Interoperability**
  - **Structural Interoperability**
  - **Syntactic Interoperability**
Triangle of Signification

Intension: Description, Property, etc.

Real (& Possible) World Referents

Terms

“Joe” + “Montana”

Syntax: Symbols

Concepts

Sense

Semantics: Meaning

Reference/ Denotation

Extension: The things that satisfy the description, property, etc.

Pragmatics: Use
**Term vs. Concept**

- **Term (terminology):**
  - Natural language words or phrases that act as indices to the underlying meaning, i.e., the concept (or composition of concepts)
  - The syntax (e.g., string) that stands in for or is used to indicate the semantics (meaning)

- **Concept:**
  - A unit of semantics (meaning), the node (entity) or link (relation) in the mental or knowledge representation model

![Diagram showing term and concept relations]
Tree vs. Graph

Directed Acyclic Graph

Root

Directed Edge

Node

Tree

Directed Cyclic Graph

Copyright © Leo Obrst, MITRE, 2002-09
Example: Metadata Registry/Repository – Contains Objects + Classification

Data Objects:
- Data Element
- Data Attribute
- Data Value
- Documents

Classification Objects:
- Ontology
- Conceptual Model
- Thesaurus
- Keyword List
- Data Schema
- XML Schema
- Taxonomy

Terminology Objects:
- Term (can be multi-lingual)
- Namespace

Meaning Objects:
- Concept
- Class
- Property
- Privileged Taxonomic Relation
- Relation
- Attribute
- Value
- Instance
Universal Core (UCore), Common Cores, Community of Interest (COI) Vocabularies

- UCore
- C2Core
- IntelCore
- BusinessCore
- COI

Upper Vocabulary
Middle Vocabularies (proposed)
Domain Vocabularies
Taxonomy: Definition

• Taxonomy:
  – A way of classifying or categorizing a set of things, i.e., a classification in the form of a hierarchy (tree)

• IT Taxonomy:
  – The classification of information entities in the form of a hierarchy (tree), according to the presumed relationships of the real world entities which they represent

• Therefore: A taxonomy is a semantic (term or concept) hierarchy in which information entities are related by either:
  – The subclassification of relation (weak taxonomies) or
  – The subclass of relation (strong taxonomies) for concepts or the narrower than relation (thesauri) for terms
  – Only the subclass/narrower than relation is a subsumption (generalization/specialization) relation
  – Subsumption (generalization/specialization) relation: the mathematical subset relation
  – Mathematically, strong taxonomies, thesauri, conceptual models, and logical theories are minimally Partially Ordered Sets (posets), i.e., they are ordered by the subset relation
    • They may be mathematically something stronger (conceptual models and logical theories)
Taxonomies: Weak

- No consistent semantics for parent-child relationship: arbitrary Subclassification Relation

- NOT a generalization / specialization taxonomy

Example: Your Folder/Directory Structure

Example: UNSPSC

<table>
<thead>
<tr>
<th>Segment</th>
<th>Family</th>
<th>Class</th>
<th>Commodity</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>Live Plant and Animal Material and Accessories and Supplies</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>00</td>
<td>00</td>
<td>Live animals</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>15</td>
<td>00</td>
<td>Livestock</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>15</td>
<td>01</td>
<td>Cats</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>15</td>
<td>02</td>
<td>Dogs</td>
</tr>
</tbody>
</table>
Taxonomies: Strong

• Consistent semantics for parent-child relationship: *Narrower than (terms) or Subclass (concepts)* Relation

• A generalization/specialization taxonomy

• For concepts: Each information entity is distinguished by a property of the entity that makes it unique as a subclass of its parent entity (a synonym for property is attribute or quality)

• For terms: each child term implicitly refers to a concept which is the subset of the concept referred to by its parent term

• What are the distinguishing properties between these three hammers?
  – Form (physical property)
  – Function (functional property)

  “Purpose proposes property” (form follows function) – for human artifacts, at least
Two Examples of Strong Taxonomies

Many representations of trees

Simple HR Taxonomy

- Person
- Organization
- Manager
- Employee
- Agent
- Animate object

Subclass of

Linnaeus Biological Taxonomy

- Kingdom: Animalia
- Phylum: Chordata
- Subphylum: Vertebrata

Class: Mammalia

- Subclass: Theria
- Infraclass: Eutheria
- Order: Primates
- Suborder: Anthropoidea
- Superfamily: Hominoidea
- Family: Hominidae
- Genus: Homo
- Species: sapiens

Class: Diapsida (Reptiles, Dinosaurs, Birds)
Another, mostly strong Taxonomy: Dewey Decimal System

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Natural sciences &amp; mathematics</td>
<td>550</td>
<td>Earth sciences</td>
</tr>
<tr>
<td>501</td>
<td>Philosophy &amp; theory</td>
<td>551</td>
<td>Geology, hydrology, meteorology</td>
</tr>
<tr>
<td>502</td>
<td>Miscellany</td>
<td>552</td>
<td>Petrology</td>
</tr>
<tr>
<td>503</td>
<td>Dictionaries &amp; encyclopedias</td>
<td>553</td>
<td>Economic geology</td>
</tr>
<tr>
<td>504</td>
<td>Not assigned or no longer used</td>
<td>554</td>
<td>Earth sciences of Europe</td>
</tr>
<tr>
<td>505</td>
<td>Serial publications</td>
<td>555</td>
<td>Earth sciences of Asia</td>
</tr>
<tr>
<td>506</td>
<td>Organizations &amp; management</td>
<td>556</td>
<td>Earth sciences of Africa</td>
</tr>
<tr>
<td>507</td>
<td>Education, research, related topics</td>
<td>557</td>
<td>Earth sciences of North America</td>
</tr>
<tr>
<td>508</td>
<td>Natural history</td>
<td>558</td>
<td>Earth sciences of South America</td>
</tr>
<tr>
<td>509</td>
<td>Historical, areas, persons treatment</td>
<td>559</td>
<td>Earth sciences of other areas</td>
</tr>
<tr>
<td>510</td>
<td>Mathematics</td>
<td>560</td>
<td>Paleontology Paleozoology</td>
</tr>
<tr>
<td>511</td>
<td>General principles</td>
<td>561</td>
<td>Paleobotany</td>
</tr>
<tr>
<td>512</td>
<td>Algebra &amp; number theory</td>
<td>562</td>
<td>Fossil invertebrates</td>
</tr>
<tr>
<td>513</td>
<td>Arithmetic</td>
<td>563</td>
<td>Fossil primitive phyla</td>
</tr>
<tr>
<td>514</td>
<td>Topology</td>
<td>564</td>
<td>Fossil Mollusca &amp; Molluscoidea</td>
</tr>
<tr>
<td>515</td>
<td>Analysis</td>
<td>565</td>
<td>Other fossil invertebrates</td>
</tr>
<tr>
<td>516</td>
<td>Geometry</td>
<td>566</td>
<td>Fossil Vertebrata (Fossil Cranata)</td>
</tr>
<tr>
<td>517</td>
<td>Not assigned or no longer used</td>
<td>567</td>
<td>Fossil cold-blooded vertebrates</td>
</tr>
<tr>
<td>518</td>
<td>Not assigned or no longer used</td>
<td>568</td>
<td>Fossil Aves (Fossil birds)</td>
</tr>
<tr>
<td>519</td>
<td>Probabilities &amp; applied mathematics</td>
<td>569</td>
<td>Fossil Mammalia</td>
</tr>
<tr>
<td>520</td>
<td>Astronomy &amp; allied sciences</td>
<td>570</td>
<td>Life sciences</td>
</tr>
<tr>
<td>521</td>
<td>Celestial mechanics</td>
<td>571</td>
<td>Not assigned or no longer used</td>
</tr>
<tr>
<td>522</td>
<td>Techniques, equipment, materials</td>
<td>572</td>
<td>Human races</td>
</tr>
<tr>
<td>523</td>
<td>Specific celestial bodies &amp;</td>
<td>573</td>
<td>Physical anthropology</td>
</tr>
<tr>
<td></td>
<td>&amp;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>525</td>
<td>Earth (Astronomical geography)</td>
<td>575</td>
<td>Evolution &amp; genetics</td>
</tr>
<tr>
<td>526</td>
<td>Mathematical geography</td>
<td>576</td>
<td>Microbiology</td>
</tr>
<tr>
<td>527</td>
<td>Celestial navigation</td>
<td>577</td>
<td>General nature of life</td>
</tr>
<tr>
<td>528</td>
<td>Ephemerides</td>
<td>578</td>
<td>Microscopy in biology</td>
</tr>
<tr>
<td>529</td>
<td>Chronology</td>
<td>579</td>
<td>Collection and preservation</td>
</tr>
<tr>
<td>530</td>
<td>Physics</td>
<td>580</td>
<td>Botanical sciences</td>
</tr>
<tr>
<td>531</td>
<td>Classical mechanics Solid mechanics</td>
<td>581</td>
<td>Botany</td>
</tr>
<tr>
<td>532</td>
<td>Fluid mechanics Liquid mechanics</td>
<td>582</td>
<td>Spermatophyta (Seed-bearing plants)</td>
</tr>
<tr>
<td>533</td>
<td>Gas mechanics</td>
<td>583</td>
<td>Dicotyledones</td>
</tr>
<tr>
<td>534</td>
<td>Sound &amp; related vibrations</td>
<td>584</td>
<td>Monocotyledones</td>
</tr>
<tr>
<td>535</td>
<td>Light &amp; parathetic phenomena</td>
<td>585</td>
<td>Gymnospermae (Pinophyta)</td>
</tr>
<tr>
<td>536</td>
<td>Heat</td>
<td>586</td>
<td>Cryptogamia (Seedless plants)</td>
</tr>
<tr>
<td>537</td>
<td>Electricity &amp; electronics</td>
<td>587</td>
<td>Pteridophyta (Vascular cryptograms)</td>
</tr>
<tr>
<td>538</td>
<td>Magnetism</td>
<td>588</td>
<td>Bryophyta</td>
</tr>
<tr>
<td>539</td>
<td>Modern physics</td>
<td>589</td>
<td>Thalloidionia &amp; Prokarvotae</td>
</tr>
<tr>
<td>540</td>
<td>Chemistry &amp; allied sciences</td>
<td>590</td>
<td>Zoological sciences</td>
</tr>
<tr>
<td>541</td>
<td>Physical &amp; theoretical chemistry</td>
<td>591</td>
<td>Zoology</td>
</tr>
<tr>
<td>542</td>
<td>Techniques, equipment, materials</td>
<td>592</td>
<td>Invertebrates</td>
</tr>
<tr>
<td>543</td>
<td>Analytical chemistry</td>
<td>593</td>
<td>Protozoa, Echinodermata, related phyla</td>
</tr>
<tr>
<td>544</td>
<td>Qualitative analysis</td>
<td>594</td>
<td>Mollusca &amp; Molluscoidea</td>
</tr>
<tr>
<td>545</td>
<td>Quantitative analysis</td>
<td>595</td>
<td>Other invertebrates</td>
</tr>
</tbody>
</table>
When is a Taxonomy enough?

• **Weak taxonomy:**
  – When you want *semantically arbitrary parent-child term or concept relations*, when the *subclassification relation* is enough
  – I.e., sometimes you just want users to navigate down a hierarchy for your specific purposes, e.g., a quasi-menu system where you want them to see locally (low in the taxonomy) what you had already displayed high in the taxonomy
  – Application-oriented taxonomies are like this
  – Then, in general, you are using weak *term* relations because the nodes are not really meant to be *concepts*, but only words or phrases that will be significant to the user or you as a classification devise

• **Strong taxonomy:**
  – When you really want to use the semantically consistent *narrower-than* (terms) or *subclass* (concepts) *relation* (a true subsumption or subset relation)
  – When you want to partition your general conceptual space
  – When you want individual conceptual buckets
  – **Note:** the subclass relation only applies to concepts; it is not equivalent (but is similar) to the narrower-than relation that applies to terms in thesauri

• **You need more than a taxonomy if you need to either:**
  – Using narrower than relation: Define term synonyms and cross-references to other associated terms, or
  – Using subclass relation: Define properties, attributes and values, relations, constraints, rules, on concepts
Take Break!
Part 2: Thesauri, Conceptual Models, & Logical Theories (Strong Ontologies)
Thesaurus: Definition

• From ANSI INISO 239.19-1993, (Revision of 239.194980):
  – A **thesaurus is a controlled vocabulary** arranged in a known order and structured so that equivalence, homographic, hierarchical, and associative relationships among terms are displayed clearly and identified by standardized relationship indicators
  – The **primary purposes of a thesaurus** are to facilitate retrieval of documents and to achieve consistency in the indexing of written or otherwise recorded documents and other items

• **Four Term Semantic Relationships:**
  – **Equivalence**: synonymous terms
  – **Homographic**: terms spelled the same
  – **Hierarchical**: a term which is broader or narrower than another term
  – **Associative**: related term

• A consistent semantics for the hierarchical parent-child relationship: broader than, narrower than

• This hierarchical ordering is a **Subsumption (i.e., generalization/specialization)** relation

• Can view just the **narrower-than** subsumption hierarchy as a **term taxonomy**

• Unlike Strong subclass-based Taxonomy, Conceptual Model, & Logical Theory: the relation is between Terms, NOT Concepts
## Thesaural Term Relationships

<table>
<thead>
<tr>
<th>Semantic Relation</th>
<th>Definition</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Synonym</strong></td>
<td>A term X has nearly the same meaning as a term Y.</td>
<td>“Car” is a synonym for “automobile”.</td>
</tr>
<tr>
<td><strong>Similar to</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Equivalent Used For</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Homonym</strong></td>
<td>A term X is spelled the same way as a term Y, which has a different meaning</td>
<td>The “bank” which is a financial institution is a homonym for the “bank” which is the side of a river or stream.</td>
</tr>
<tr>
<td><strong>Spelled the Same Homographic</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Broader Than</strong></td>
<td>A term X is broader in meaning than a term Y.</td>
<td>“Vehicle” has a broader meaning than “automobile”.</td>
</tr>
<tr>
<td>(Hierarchic: parent of)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Narrower Than</strong></td>
<td>A term X is narrower in meaning than a term Y.</td>
<td>“Automobile” has a narrower meaning than “vehicle”.</td>
</tr>
<tr>
<td>(Hierarchic: child of)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Associated</strong></td>
<td>A term X is associated with a term Y, i.e., there is some unspecified relationship between the two.</td>
<td>A “comb” is associated with a “barber”.</td>
</tr>
<tr>
<td>** Associative Related**</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Thesaurus vs. Ontology

Controlled Vocabulary

Terms: Metal working machinery, equipment and supplies, metal-cutting machinery, metal-turning equipment, metal-milling equipment, milling insert, turning insert, etc.

Relations: use, used-for, broader-term, narrower-term, related-term

Terms: Metal working machinery, equipment and supplies, metal-cutting machinery, metal-turning equipment, metal-milling equipment, milling insert, turning insert, etc.

Relations: subclass-of; instance-of; part-of; has-geometry; performs, used-on; etc.

Properties: geometry; material; length; operation; UN/SPSC-code; ISO-code; etc.

Values: 1; 2; 3; “2.5 inches”; “85-degree-diamond”; “231716”; “boring”; “drilling”; etc.

Axioms/Rules: If milling-insert(X) & operation(Y) & material(Z)=HG_Steel & performs(X, Y, Z), then has-geometry(X, 85-degree-diamond).

Ontology

Logical-Conceptual Semantics (Strong)

Real (& Possible) World Referents

Logical Concepts

‘Semantic’ Relations:
- Equivalent =
- Used For (Synonym) UF
- Broader Term BT
- Narrower Term NT
- Related Term RT

Semantic Relations:
- Subclass Of
- Part Of
- Arbitrary Relations
- Meta-Properties on Relations
Moving target indicators

- Aerial imagery
- Radar imagery
- Radar photography

Imagery

- Infrared imagery

Imaging Systems

- Imaging radar

Combat support equipment

Intelligence and electronic warfare equipment

Narrower than

Related to
When is a Thesaurus enough?

- When you don’t need to define the concepts of your model, but only the terms that refer to those concepts, i.e., to at least partially index those concepts
- Ok, what does that mean?
- If you need an ordered list of terms and their synonyms and loose connections to other terms (cross-references)
- Examples:
  - If you need to use term buckets (sets or subsets) to use for term expansion in a keyword-based search engine
  - If you need a term classification index for a registry/repository, to guarantee uniqueness of terms and synonyms within a Community of Interest or namespace that might point to/index a concept node
- **You need more than a thesaurus** if you need to define properties, attributes and values, relations, constraints, rules, on concepts
  - You need either a **conceptual model** (weak ontology) or a **logical theory** (strong ontology)
Conceptual Models: Weak Ontologies

- Many conceptual domains cannot be expressed adequately with a taxonomy (nor with a thesaurus, which models term relationships, as opposed to concept relationships)
- Conceptual models seek to model a portion of a domain that a database must contain data for or a system (or, recently, enterprise) must perform work for, by providing users with the type of functionality they require in that domain
- UML is paradigmatic modeling language
- Drawbacks:
  - Models mostly used for documentation, required human semantic interpretation
  - Limited machine usability because cannot directly interpret semantically
  - Primary reason: there is no Logic that UML is based on
- **You need more than a Conceptual Model** if you need machine-interpretability (more than machine-processing)
  - You need a logical theory (high-end ontology)
Conceptual Model: UML Example

Human Resource Conceptual Model
Logical Theories: Strong Ontologies

• Can be either Frame-based or Axiomatic
  – **Frame-based**: node-and-link structured in languages which hide the logical expressions, entity-centric, like object-oriented modeling, centering on the entity class, its attributes, properties, relations/associations, and constraints/rules
  – **Axiomatic**: axiom/rule-structured in languages which expose the logical expressions, non-entity-centric, so axioms that refer to entities (classes, instances, their attributes, properties, relations, constraint/rules) can be distributed
Logical Theories: More Formally

Conceptualization $C$

Language $L$

Models $M(L)$

Ontology

Intended models $I_{M(L)}$

A More Complex Picture (from E-Commerce)
Axioms, Inference Rules, Theorems, Theory

1. Theorems are licensed by a valid proof using *inference rules* such as Modus Ponens.

2. Theorems proven to be true can be added back in, to be acted on subsequently like axioms by inference rules.

3. Possible other theorems (as yet unproven).

4. Ever expanding theory.
<table>
<thead>
<tr>
<th>Axioms</th>
<th>Inference Rules</th>
<th>Theorems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class(Thing)</td>
<td>And-introduction: given P, Q, it is valid to infer P (\land) Q.</td>
<td>If P (\land) Q are true, then so is P (\lor) Q.</td>
</tr>
<tr>
<td>Class(Person)</td>
<td>Or-introduction: given P, it is valid to infer P (\lor) Q.</td>
<td>If X is a member of Class(Parent), then X is a member of Class(Person).</td>
</tr>
<tr>
<td>Class(Parent)</td>
<td>And-elimination: given P (\land) Q, it is valid to infer P.</td>
<td>If X is a member of Class(Child), then X is a member of Class(Person).</td>
</tr>
<tr>
<td>Class(Child)</td>
<td>Excluded middle: P (\lor) (\neg)P (i.e., either something is true or its negation is true)</td>
<td>If X is a member of Class(Child), then NameOf(X, Y) and Y is a String.</td>
</tr>
<tr>
<td>If SubClass(X, Y) then X is a subset of Y. This also means that if A is a member of Class(X), then A is a member of Class(Y)</td>
<td>Modus Ponens: given P (\rightarrow) Q, P, it is valid to infer Q</td>
<td>If Person(JohnSmith), then (\neg) ParentOf(JohnSmith, JohnSmith).</td>
</tr>
<tr>
<td>SubClass(Person, Thing)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SubClass(Parent, Person)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SubClass(Child, Person)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ParentOf(Parent, Child)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NameOf(Person, String)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AgeOf(Person, Integer)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If X is a member of Class(Parent) and Y is a member of Class(Child), then (\neg) (X = Y)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Ontology Representation Levels

<table>
<thead>
<tr>
<th>Level</th>
<th>Example Constructs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Knowledge Representation (KR) Language (Ontology Language) Level</strong>: Meta Level to the Ontology Concept Level</td>
<td>Class, Relation, Instance, Function, Attribute, Property, Constraint, Axiom, Rule</td>
</tr>
<tr>
<td><strong>Ontology Concept (OC) Level</strong>: Object Level to the KR Language Level, Meta Level to the Instance Level</td>
<td>Person, Location, Event, Parent, Hammer, River, FinancialTransaction, BuyingAHouse, Automobile, TravelPlanning, etc.</td>
</tr>
<tr>
<td><strong>Ontology Instance (OI) Level</strong>: Object Level to the Ontology Concept Level</td>
<td>Harry X. Landsford III, Ralph Waldo Emerson, Person560234, PurchaseOrderTransactionEvent6117090, 1995-96 V-6 Ford Taurus 244/4.0 Aerostar Automatic with Block Casting # 95TM-AB and Head Casting 95TM</td>
</tr>
</tbody>
</table>

**Language**
- **Ontology (General)**
- **Knowledge Base (Particular)**
**Ontology Example from Electronic Commerce: the general domain of machine tooling & manufacturing; note that these are expressed in English, but usually would be in expressed in a logic-based language**

<table>
<thead>
<tr>
<th>Concept</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Classes (general things)</strong></td>
<td>Metal working machinery, equipment and supplies, metal-cutting machinery, metal-turning equipment, metal-milling equipment, milling insert, turning insert, etc.</td>
</tr>
<tr>
<td><strong>Instances (particular things)</strong></td>
<td>An instance of metal-cutting machinery is the “OKK KCV 600 15L Vertical Spindle Direction, 1530x640x640mm 60.24&quot;x25.20&quot;x25.20 X-Y-Z Travels Coordinates, 30 Magazine Capacity, 50 Spindle Taper, 20kg 44 lbs Max Tool Weight, 1500 kg 3307 lbs Max Loadable Weight on Table, 27,600 lbs Machine Weight, CNC Vertical Machining Center”</td>
</tr>
<tr>
<td><strong>Relations: subclass-of, (kind_of), instance-of, part-of, has-geometry, performs, used-on, etc.</strong></td>
<td>A kind of metal working machinery is metal cutting machinery, A kind of metal cutting machinery is milling insert.</td>
</tr>
<tr>
<td><strong>Properties</strong></td>
<td>Geometry, material, length, operation, ISO-code, etc.</td>
</tr>
<tr>
<td><strong>Values:</strong></td>
<td>1; 2; 3; “2.5”, inches”; “85-degree-diamond”; “231716”; “boring”; “drilling”; etc.</td>
</tr>
<tr>
<td><strong>Rules (constraints, axioms)</strong></td>
<td>If milling-insert(X) &amp; operation(Y) &amp; material(Z)=HG_Steel &amp; performs(X, Y, Z), then has-geometry(X, 85-degree-diamond). [Meaning: if you need to do milling on High Grade Steel, then you need to use a milling insert (blade) which has a 85-degree diamond shape.]</td>
</tr>
</tbody>
</table>
Example: Inference and Proof

Proof Using Inference Rule of Modus Ponens

**Given:** If motherOf is a subProperty of parentOf, and Mary is the mother of Bill, then Mary is the parentOf Bill

1. motherOf is a subProperty of parentOf
2. Mary is the motherOf Bill

**Infer:** Mary is the parentOf Bill

**Deduction** A method of reasoning by which one infers a conclusion from a set of sentences by employing the axioms and rules of inference for a given logical system.

Ontology/KR
Expressible as Language and Graph

- In ontology and knowledge bases, nodes are predicate, rule, variable, constant symbols, hence graph-based indexing, viewing
- Links are connections between these symbols: Semantic Net!

CYC MELD Expression Example

What’s important is the logic!
Upper, Middle, Domain Ontologies

Most General Thing
- Time
- Identity
- Space
- Material
- Processes
- Locations
- Organizations
- Facilities
- People

Upper Ontology (Generic Common Knowledge)
- Terrorist
- Org
- Financier
- Terrorist Org
- Al Qaeda

Middle Ontology (Domain-spanning Knowledge)
- Jihadist Terrorist

Lower Ontology (individual domains)

Lowest Ontology (sub-domains)

Areas of Interest

But Also These!
Ontology Content Architecture: More Complex View

Abstract Top Ontology Layer (Set Theory, Category Theory)*

Knowledge Representation Language Layer (Abstract Core Ontology)*

Ontology Universal (Class) Layer

Ontology Individual (Instance) Layer

Epistemological Data Layer: Schema + Tuples

Ontology Universals & Individuals Layer: Upper, Mid-Level, Domain Ontologies

Ontology Lifecycle

1) Rationale: Why do you need an ontology?

2) Analysis 1 (Competency Questions)
   - Bottom-Up: What are semantics of current data sources?
   - Top-Down: What would you like to ask?

3) Analysis 2
   - What are the referents, concepts: entities, relations, properties, rules?
   - What are the terms that index the referents: terminology?

4) Analysis 3
   - What are the resources available to harvest: vocabularies, schemas, taxonomies, conceptual models, ontologies?
   - Are there domain standards, upper/middle ontologies to embed what we create within?

5) Design 1
   - What ontology architecture do we choose?
   - How expressive is the ontology language we need?
   - What conceptualization?
   - How do we model these entities, relations, properties, rules?
   - What are the instances of these?
   - What data sources mappings can link to these? How?
   - What kinds of ontology tools do we need?

6) Implement 1
   - Implement the ontology server we will need: periodicity, granularity, configuration management
   - Implement the infrastructure, services of our architecture: enhance the server with application, SOA support

7) Design 2
   - Are we done with ontology development?

8) Analysis 4
   - Refine with domain experts, end users

9) Design 3
   - Refine conceptualization

10) Implement 2
    - Refine ontology

11) Deploy 1
    - Provide ontology application services

12) Deploy 2
    - Correct problems

13) Analysis 5
    - Interrogate users
    - Refine reqs
    - More resources?

14) Design 4
    - How can changes needed be made?
    - Refine reqs

15) Analysis 6
    - Interrogate users
    - Refine reqs

16) Design 5
    - How can changes needed be made?
    - Refine reqs

17) Analysis 7
    - Interrogate users
    - Refine reqs

18) Design 6
    - How can changes needed be made?
    - Refine reqs
Ontology Maturity Model

OMM Level 1
- Least Mature
- Mainstream syntactic/structural DB technology (+ data warehouses + data marts), unstructured data addressed by procedural information extraction, no persistent linkage of semantics to syntax/structure, ad hoc local semantics sometimes captured in data dictionary & commented in extraneous code; no clear distinction made between term & concept (referent)

OMM Level 2
- OMM Level 4
- Consipeced, consistent local semantics captured, some real domain semantics represented as persistent & maintained models (local ontologies); term & concept (referent) distinguished; databases and information extraction routines use local ontologies

OMM Level 3
- Focus is on capture of real domain semantics, mostly represented as persistent & maintained models (frame ontologies); term resources linked to models; database and information extraction routines use some domain ontologies

OMM Level 4
- OMM Level 5
- Consistent & pervasive capture of real domain semantics, represented as persistent & maintained models (frame ontologies, some axioms); some linkage to upper/middle; some inference supported;

OMM Level 5
- Most Mature
- Consistent, pervasive capture of real domain semantics embedded under common middle/upper semantics (axiomatized ontologies); extensive inference

From less to more mature

Ontology Spectrum

- Scope
- KR Construct
- Parent-Child Relation
- Machine Processing

Term
- Concept
- Taxonomy
- Thesaurus
- Ontology

Classifications:
- Weak Taxonomy
- Strong Taxonomy
- Conceptual Model (weak ontology)
- Logical Theory (strong ontology)

Relations:
- Sub-classification of
- Narrower Than
- SubClass of
- Disjoint SubClass of with Transitivity, etc.
Ontology Spectrum: Complexity of Applications

- Concept (referent category) based
- Term - based
- Thesaurus
- Taxonomy
- Ontology
- Logical Theory

More Expressive Semantic Models Enable More Complex Applications

- Categorization, Simple Search & Navigation, Simple Indexing
- Synonyms, Enhanced Search (Improved Recall) & Navigation, Cross Indexing
- Enterprise Modeling (system, service, data), Question-Answering (Improved Precision), Querying, SW Services
- Real World Domain Modeling, Semantic Search (using concepts, properties, relations, rules), Machine Interpretability (M2M, M2H semantic interoperability), Automated Reasoning, SW Services
# Recall and Precision

<table>
<thead>
<tr>
<th>Recall</th>
<th>The percentage of relevant documents retrieved</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Calculation:</strong></td>
<td>Number of relevant docs retrieved</td>
</tr>
<tr>
<td></td>
<td>Number of relevant docs</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Precision</th>
<th>The percentage of retrieved documents judged relevant</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Calculation:</strong></td>
<td>Number of relevant docs retrieved</td>
</tr>
<tr>
<td></td>
<td>Number of docs retrieved</td>
</tr>
</tbody>
</table>

**Available Data**

**Recall**

**Precision**
What Problems Do Ontologies Help Solve?

• Heterogeneous database problem
  – Different organizational units, Service Needers/Providers have radically different databases
  – Different syntactically: what’s the format?
  – Different structurally: how are they structured?
  – Different semantically: what do they mean?
  – They all speak different languages

• Enterprise-wide system interoperability problem
  – Currently: system-of-systems, vertical stovepipes
  – Ontologies act as conceptual model representing enterprise consensus semantics
  – Well-defined, sound, consistent, extensible, reusable, modular models

• Relevant document retrieval/question-answering problem
  – What is the meaning of your query?
  – What is the meaning of documents that would satisfy your query?
  – Can you obtain only meaningful, relevant documents?
A Business Example of Ontology

![Ontology Diagram]

- **Catalog No.**
- **Shape**
- **Size (in)**
- **Price ($US)**

<table>
<thead>
<tr>
<th>Part No.</th>
<th>Geom</th>
<th>Diam (mm)</th>
<th>Price ($US)</th>
</tr>
</thead>
<tbody>
<tr>
<td>XAB023</td>
<td>R</td>
<td>37</td>
<td>.35</td>
</tr>
<tr>
<td>XAB035</td>
<td>S</td>
<td>31</td>
<td>.45</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Mfr No.</th>
<th>Shape</th>
<th>Size (in)</th>
<th>Price ($US)</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-Machina</td>
<td>550296</td>
<td>Round</td>
<td>1.5</td>
<td>.35</td>
</tr>
<tr>
<td>iMetal Corp.</td>
<td>XAB023</td>
<td>Round</td>
<td>1.5</td>
<td>.75</td>
</tr>
<tr>
<td>E-Machina</td>
<td>550298</td>
<td>Square</td>
<td>1.25</td>
<td>.45</td>
</tr>
<tr>
<td>iMetal Corp.</td>
<td>XAB035</td>
<td>Square</td>
<td>1.25</td>
<td>.25</td>
</tr>
</tbody>
</table>

- **Supplier A**
- **Supplier B**
- **Buyer**
A Military Example of Ontology

<table>
<thead>
<tr>
<th>Tid</th>
<th>Type</th>
<th>Long Lat</th>
<th>T stamp</th>
<th>S-code</th>
<th>Model</th>
<th>Coord</th>
<th>Sense</th>
<th>Time</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNM023</td>
<td>MIG-29</td>
<td>121.135°</td>
<td>13458</td>
<td>330298</td>
<td>F-14D</td>
<td>121°8'6&quot;</td>
<td>2.35</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CNM035</td>
<td>Tupolev TU154</td>
<td>121.25°</td>
<td>13465</td>
<td>330298</td>
<td>AH-1G C</td>
<td>121°2'2&quot;</td>
<td>2.45</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>Identifier</th>
<th>Signature</th>
<th>Location</th>
<th>Time Observed</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Army</td>
<td>330295</td>
<td>F-14D</td>
<td>121°8'6&quot;</td>
<td>2.35</td>
<td></td>
</tr>
<tr>
<td>Army</td>
<td>CNM023</td>
<td>MIG-29</td>
<td>121.135°</td>
<td>13458</td>
<td></td>
</tr>
<tr>
<td>Navy</td>
<td>330298</td>
<td>AH-1G C</td>
<td>121°2'2&quot;</td>
<td>2.45</td>
<td></td>
</tr>
<tr>
<td>Army</td>
<td>CNM035</td>
<td>Tupolev TU154</td>
<td>121.25°</td>
<td>13465</td>
<td></td>
</tr>
</tbody>
</table>

Ontology

Identifier

Signature

Location

Time Observed

Commander, S2, S3

Copyright © Leo Obrst, MITRE, 2002-09
Ontologies & the Data Integration Problem

- DBs provide generality of storage and efficient access
- Formal data model of databases insufficiently semantically expressive
- The process of developing a database discards meaning
  - Conceptual model $\rightarrow$ Logical Model $\rightarrow$ Physical Model
  - Keys signify some relation, but no solid semantics
  - DB Semantics = Schema + Business Rules + Application Code
- Ontologies can represent the rich common semantics that spans DBs
  - Link the different structures
  - Establish semantic properties of data
  - Provide mappings across data based on meaning
  - Also capture the rest of the meaning of data:
    - Enterprise rules
    - Application code (the inextricable semantics)
Complexity of Semantic Integration with/without Ontologies

- An ontology allows for near linear semantic integration (actually \(2n-1\)) rather than near \(n^2\) (actually \(n^2 - n\)) integration
  - Each application/database maps to the "lingua franca" of the ontology, rather than to each other

**Ordinary Integration: \(N^2\)**

- A ↔ B
- A ↔ C
- B ↔ C

Add D:

- A ↔ D
- B ↔ D
- C ↔ D

**Ontology Integration: \(N\)**

- B ↔ A ↔ C
- 2 Nodes: 2 Edges
- 3 Nodes: 6 Edges
- 4 Nodes: 12 Edges
- 5 Nodes: 20 Edges

Add D:

- A ↔ D
- A ↔ D
Approximate Cost/Benefit of Moving up the Ontology Spectrum

- **Cost**
  - Taxonomy
  - Thesaurus
  - Conceptual Model
  - Logical Theory

- **Benefit**
  - Increasingly greater benefit because of increased semantic interoperability, precision, level machine-human interaction

- **Cost**
  - Higher initial costs at each step up

- **Benefit**
  - Much lower eventual costs because of reuse, less analyst labor
Parts 1 & 2 Conclusions

- Ontology: a specification of a conceptualization, vocabulary + model, theory

- Informally, ontology and model are taken to be synonymous, i.e., a description of the structure and meaning of a domain, a conceptual model

- **Bottom Line**: an Ontology models Concepts, i.e., the entities (usually structured in a class hierarchy with multiple inheritance), relations, properties (attributes), values, instances, constraints, and rules used to model one or more domains
  1) A logical theory
  2) About the world or some portion of the world
  3) Represented in a form semantically interpretable by computer
  4) Thus enabling automated reasoning comparable to a human’s

- Logically, you can view an ontology as a set of Axioms (statements and constraints/rules) about some domain

- Using the axioms and some defined Inference Rules (example: Modus Ponens), you can derive (prove true) Theorems about that domain, and thus derive knew knowledge
Lunch!
Agenda, Part 3a: Knowledge Representation
26 Years: Knowledge Representation & Engineering Research Threads Converging*

*And 20 yrs more before this!
What is Knowledge Representation?

• Principles of KR: Davis, Shrobe, Szolovits (1993)*:
  – A KR is a surrogate for real things, events, relationships
  – A KR is a set of ontological commitments, a model for a particular conception of the world
  – A KR is a partial theory of intelligent reasoning
  – A KR is a medium for efficient computation
  – A KR is a medium of human expression

• Principled, Declarative, Modular, Reusable: Represent Once!

• Other issues:
  – Design & development vs. runtime implementation/use
  – Knowledge partitioning and compilation
  – Representation tightly coupled with Reasoning Methods: the Language determines the Reasoning

Evolution of KR

- KR derived from semantic networks of 60s-70s, Quillian, 1968; Minsky, 1975; Brachman, 1978
- Brachman & Levesque, 1985: survey of newer semantic nets, frame-based languages: KL-ONE (Brachman & Schmolze, 1985)
- Increasing formalization, logicization: SIGART bulletin 2:3, 1991: seminal encapsulation of state of the art
- Development of a sound theoretical basis for the syntax, semantics, and inference methods employed: DLs
- DARPA Knowledge Sharing Initiative (KSI, early 90s): Knowledge Interchange Format (KIF), Ontolingua, Generic Frame Protocol (GFK), rise of Ontological Engineering
- DARPA High Performance Knowledge Bases (HPKB), Rapid Knowledge Formation (RKF) (late 90s): Open Knowledge Base Connectivity (OKBC) language
- DARPA Agent Markup Language (DAML) (early 2000s): DAML+ OIL, OWL
Knowledge Representation and Related Disciplines

- Mathematics
- Logic
- Philosophy
- Formal Methods
- Formal Semantics
- Ontology
- Informal Ontology
- Artificial Intelligence
- Informal Ontology
- Enterprise Engineering
- Industrial Engineering
- Linguistics
- Business Management
- Computer Science
- Database Theory
- Sociology
- Formal Ontology
- Knowledge Management
- Knowledge Representation
- Ontology Engineering
- Knowledge Engineering
- Conceptual Modeling
- Software/Data Engineering
- Object Modeling

Copyright © Leo Obrst, MITRE, 2002-09
Semantic Networks

“‘A semantic network is a graph structure in which nodes (or vertices) represent concepts, while the arcs between these nodes represent relations among concepts.’”

- Semantic Networks were not formally defined
- Reasoning methods were based on implementation strategies, not on formal language
- First formalization based on logic: the “frame” language KL-ONE
Expert Systems & Their Problems

- Based on “production rules” using the Rete Algorithm:
  - **Condition-Action** (antecedent/consequent) Rules: If Conditions $\alpha$ hold, then execute Actions $\beta$,
  - where $\alpha$ are predicates true of the state of the environment at time of rule-firing (e.g., “AND $<$temperature $\geq$ 212 degrees$>$, $<$oil_flowing = true$>$)
  - and $\beta$ are actions such as “push rule 14 onto Agenda”, or set “AlertMonitor = true”, etc., which thereby changes the state of the environment, allowing other rules to prospectively fire (if their conditions are met)
  - **Forward Chaining**: go from state of the world and see which conditions of which rules match that state, firing off rules that apply
  - **Backward Chaining**: start at a rule’s goal (the theorem to be proved true), assume it to be true, then its antecedent conditions would generate new goals, with the new goals matching the consequents of other rules

- All knowledge is represented at same level: non-modular, non-reusable, unmaintainable
- Undebuggable when complex, non-deterministic rule-firings
- Experts don’t necessarily have insight into “how” they know things
- Everyone speaks a natural language, but few can describe the properties of a natural language (coherently, consistently)
Frame Languages

• Frame-based systems are KR systems that use frames
• Introduced by Marvin Minsky (1975) to represent domain knowledge
  – Represent a stereotypical situation
  – Way of structuring knowledge
  – A network of nodes and relations
  – Generic (nonterminal) knowledge bottoming out in instances (terminals)
• The notion of a frame corresponds to early LISP programming language terminology: slot & filler, record-based, defstruct-like
• Frames represent Concepts, have additional information attached to them: definitional, how to use, etc
• In frame terminology, a concept is a Class, and a relation is a Slot
• Attributes (sometimes called properties) are just slots defined on a domain (a specific class subtree) or one of its subdomains (a subclass of a domain class).
• Frames are close to the OO Paradigm: i.e., they are object-centered (entity or class-centered)
• First formalized frame KR language: KL-One
• Bottom Line: Frames are equivalent to a Logical Representation
Frame Languages: Example

- (defineClass StationWagon
  (superclass Automobile)
  (doors 5)
  (model *noDefault*)
  (manufacturer Manufacturer))

- (defineInstance inst-345678
  (class StationWagon)
  (doors 3)
  (model Taurus)
  (manufacturer Ford)
  (weight WeightMeasure))

- Can have multiple parents
- Inheritance of slots (relations, attributes):
  - SubClass (isa) relation
  - InstanceOf relation
- Defaults & Overrides
- Define new slots
- Can view a Frame as a Type
Axiom-based (Axiomatic) KR 1

- These are based on a formalized logic
- **Typically First Order Logic (Predicate Calculus), or a subset of FOL**
  - Could be based on weaker Propositional Logic, which only represents *propositions*, i.e., expressions that are true or false
  - Examples: “It’s cold outside”, “John is sick”, “The current President of the United States is George W. Bush”
  - All of these are either true or false, or possibly unknown: “Unicorns are nice”
  - But each of these is an X which is either true or false
  - We would like to get more expressive, talk logically about individuals (instances) and predicates (relations, properties, attributes) inside the *proposition*
  - The FOL enables us to talk about instances: “Some people don’t like peaches”, i.e., *There are some X who are people and those X don’t like peaches.*

- Contain axioms, which are logical expressions asserted to be true, all the time, given what we know about the world: *All humans are mortal*
Axiom-based (Axiomatic) KR 2

- Theorems are proven by using inference rules applied to axioms:
  - Prove: John is mortal
  - Proof: If all humans are mortal, and John is a human, then John is mortal
  - Theorems, once proven, **add to the knowledge that is in your ontology model:** they generate **NEW knowledge**

- A number of threads:
  - Description Logics
  - But also Logic Programming as in Prolog
  - Cyc, KIF (Knowledge Interchange Format)
  - Theorem provers that use FOL or higher-order logic
  - RDF/S and OWL are axiom-based, though by design, they also contain *frame-based* representation. Why? To assist developers and users who know the *Object-Oriented paradigm of entity (class) centered or focused modeling*

- **Bottom-line:** an axiom-based ontology system is not object-centered like an OO modeling system, but instead has the modeling knowledge about any given object (e.g., entity or relation) **distributed** across the ontology.
Issues: Expressivity

• What do you want to do with your KR language?
  – Build an ontology, build a knowledge base
  – Check consistency of your knowledge
  – Check completeness of your knowledge
  – I.e., Model checking, model finding
  – Automatically classify new concepts, assertions
  – Query the KB (search & navigation)
  – Perform other inference (sometimes called rule-based reasoning)
    • Deduction
    • Induction
    • Abduction
  – Add probabilistic reasoning
  – Reason over beliefs (Truth Maintenance Systems), i.e., evidential reasoning
  – Have built in modal operators: necessity/possibility, obligation/permission/prohibition, temporal, etc.
Propositional & Predicate Logic

• Propositional Logic
  – Limitation: cannot speak about individuals (instances)
  – Granularity not fine enough
  – Propositions: truth-functions
    If Plato is human, then Plato is mortal \( p \rightarrow q \)
    Plato is human \( p \)
    \[
    \begin{array}{c}
    \hline
    \text{Plato is mortal} \\
    \text{q} \\
    \end{array}
    \]
    Modus Ponens

• Predicate Logic
  – Finer distinctions: can talk about individuals (instances)
    If Plato is human, then Plato is mortal \( \forall x: p(x) \rightarrow q(x) \)
    Plato is human \( p(plato) \)
    \[
    \begin{array}{c}
    \hline
    \text{Plato is mortal} \\
    \text{q(plato)} \\
    \end{array}
    \]
    Modus Ponens
  – An instantiated predicate is a proposition, e.g., human(plato) = true
Modal Logic

- Modal Logic: want to express and reason about various other kinds of states of affairs, possibility vs. necessity, etc.
  - Introduce new sentential operators (focus only on propositional modal logic)

<table>
<thead>
<tr>
<th>Logic</th>
<th>Symbols</th>
<th>Expressions Symbolized</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modal Logic</td>
<td>□</td>
<td>It is necessary that ..</td>
</tr>
<tr>
<td></td>
<td>◊</td>
<td>It is possible that ..</td>
</tr>
<tr>
<td>Deontic Logic</td>
<td>O</td>
<td>It is obligatory that ..</td>
</tr>
<tr>
<td></td>
<td>P</td>
<td>It is permitted that ..</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>It is forbidden that ..</td>
</tr>
<tr>
<td>Temporal Logic</td>
<td>G</td>
<td>It will always be the case that ..</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>It will be the case that ..</td>
</tr>
<tr>
<td></td>
<td>H</td>
<td>It has always been the case that ..</td>
</tr>
<tr>
<td></td>
<td>P</td>
<td>It was the case that ..</td>
</tr>
<tr>
<td>Doxastic Logic</td>
<td>Bx</td>
<td>x believes that ..</td>
</tr>
</tbody>
</table>

- http://plato.stanford.edu/entries/logic-modal/
## Modal Propositions & Predicates

<table>
<thead>
<tr>
<th>Modal Propositions and Predicates in English</th>
<th>Modal Propositions and Predicates in Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Necessarily, if John is an unmarried man, John is a bachelor.</td>
<td>$\Box (P \rightarrow Q)$</td>
</tr>
<tr>
<td>2) Possibly, if John likes sugar, he likes chocolate.</td>
<td>$\Diamond (P \rightarrow Q)$</td>
</tr>
<tr>
<td>3) Necessarily, an unmarried man is a bachelor.</td>
<td>$\Box (\forall x. \text{unmarriedMan}(x) \rightarrow \text{bachelor}(x))$</td>
</tr>
<tr>
<td>4) Necessarily, every human has parents.</td>
<td>$\Box (\forall x. \text{human}(x) \rightarrow \text{hasParents}(x))$</td>
</tr>
<tr>
<td>5) If a person works at a company, it’s possible that he is not an employee. (he could be a contractor, for example)</td>
<td>$\exists x. \text{person}(x) \land \text{company}(y) \land \text{worksAt}(x, y) \rightarrow \Diamond \neg \text{employeeOf}(x, y)$</td>
</tr>
</tbody>
</table>
Description Logic: Definitions

- **What is a Description Logic?** Terminological Logic, Concept Logic, based on: Concept Language, Term Subsumption Language
  - A declarative formalism for the representation and expression of knowledge and sound, tractable reasoning methods founded on a firm theoretical (logical) basis
    - DL frame-based semantic network + logic (compositional syntax and model-theoretic semantics)
    - usual logical formulation of a concept would be as a single-variable predicate, i.e., in lambda calculus, as (MacGregor, 1991):
      - adult males: \( \lambda x. \text{Male}(x) \cup \text{Adult}(x) \)
  - Expressive, sound & complete, decidable, classical semantics, tractable reasoning
  - Function-free FOL using at most 3 variables (basic)

- **A description**: an expression in a formal language that defines a set of instances or tuples

- **DL**: a syntax for constructing descriptions and a semantics that defines the meaning of each description
Description Logic: Components

- **T-box: Terminological box** – concepts, classes, predicates
  - One or more subsumption hierarchies/taxonomies of descriptions
  - Terminological axioms: introduce names of concepts, roles
  - Concepts: denote entities
  - Roles: denote properties (binary predicates, relations)
  - OO? No, but related. Why: no generally agreed upon formal basis to OO, though attempts (emerging UML)
    - Isa generalization/specialization, Top/ Bottom
    - Part-of: mereology, mereotopology (parts+connections)
    - Other relations: aggregation, etc.
  - Subsumption: comparable to matching or unification in other systems

- **A-box: Assertional box** – individuals, constants
  - Instances in the OO world, tuples in the DB world
Description Logic: Inference Methods & Properties

• Inference Methods (all based on subsumption)
  – **classification**: where do descriptions belong in hierarchies (subsumers, subsumees)
  – **detecting contradiction**: are descriptions coherent/satisfiable and is the KB consistent/satisfiable
  – **completion inference**: what are the logical consequences of axioms, inheritance

• Inference algorithms properties:
  – **soundness**: any expression that can be derived from the KB is logically implied by that KB
  – **completeness**: any expression that is logically implied by the KB can be derived
  – **decidability**: can a sound and complete algorithm be constructed?
  – **complexity**: is it tractable (worst-case polynomial time) or intractable?
  – **expressivity**: [for formal definition of expressivity of T-Box (Baader, 1990); A-Box (Speel, 1996a, p. 69)]
    • roughly: expressivity and tractability are inversely proportional
    • some expressive formalisms may be intractable or even undecidable
Example: OIL, which became DAML
+OIL, which became OWL

http://www.ontoknowledge.org/oil/TR/oil.long.html
First Order & Higher Order Logics: the basis of other Ontology Languages

• **FOL** semi-decidable
  – Decidable: there is an effective method for telling whether or not each formula of a system is a theorem of that system or not
  – Semi-decidable: If a formula really is a theorem of a system, eventually will be able to prove it is, but not if it is not: may never terminate

• **Second Order**: sometimes used in linguistics
  – “Tall”, “Most”, etc.
  – Quantification over Individual & Predicate variables
  – $\exists \phi (\phi(a) \land F(\phi))$: “John has an unusual property”

• **CYC**: MELD, CYCL, has some constrained 2nd order reasoning

• **Theorem-provers**
  – HOL, Otter, etc.

• **Prolog & Cousins**
  – Restricted FOL: Horn Clauses (only 1 un-negated term in a formula, resolution method proves the contradiction of the negation of a term)
  – Non-standard negation: negation by finite failure
  – Closed World Assumption
  – Declarative + Operational Semantics: use of Cut

• **Other**: Conceptual Graphs, UML, Expert System Shells, Modal Logics
Looking Ahead: From Ontology Spectrum to Logic Spectrum

Logic Spectrum will cover this area

Logical Theory
- Modal Logic
- First Order Logic
- Description Logic
- DAML+OIL, OWL
- UML

Conceptual Model
- Is Subclass of
- RDF/S
- XTM
- Extended ER
- ER
- Is Sub-Classification of
- DB Schemas, XML Schema
- Thesaurus
- Has Narrower Meaning Than
- Taxonomy
- Is Sub-Classification of
- Relational Model, XML

Semantic Interoperability
- Strong semantics

Structural Interoperability
- Weak semantics

Syntactic Interoperability
Logic Spectrum

- Propositional Logic (PL)
- First-Order Logic (FOL): Predicate Logic, Predicate Calculus
- Modal Propositional Logic
- Description Logics
- Modal Predicate Logic (Quantified Modal Logic)
- Second Order Logic (SOL)
- Higher Order Logic (HOL)

From less to more expressive logics
Logic Spectrum: Classical Logics: PL to HOL

First-Order Logic (FOL): Predicate Logic, Predicate Calculus

Modal Propostional Logic

Propositional Logic (PL)

Description Logics

Logic Programming (Horn Clauses)

Modal Predicate Logic (Quantified Modal Logic)

Second Order Logic (SOL)

Higher Order Logic (HOL)

most expressive

SOL + Complex Types + Higher-order Predicates (i.e., those that take one or more other predicates as arguments)

FOL + Quantifiers (∀, ∃) over Predicates

FOL + Modal operators

PL + Predicates + Functions + Individuals + Quantifiers (∀, ∃) over Individuals

Syntactic Restriction of FOL

Decidable fragments of FOL: unary predicates (concepts) & binary relations (roles) [max 3 vars]

Substructural Logics: focus on structural rules

Propositions (True/False) + Logical Connectives (¬, ∧, ∨, →, ↔)

Copyright © Leo Obrst, MITRE, 2002-09
Agenda, Part 3b: Ontological Engineering
Ontology Modeling Issues

• What do you model in? **KR Language**
  – OO Frame vs. DL or FOL Axiom?
• What do you model? **Concepts**
• Concepts:
  – Concepts “stand in for” objects in the real world (possible world)
  – Entities & relations
  – Universals & Particulars
  – Classes & Instances/Individuals
• How are Concepts modeled?
How are Concepts Modeled?

- **Meta-class, Class, Instance**
  - If have a meta-class Class, then all Classes are instances of that
  - Remember the 3 Representation Levels: Meta, Object, Instance
  - An Instance is a specific thing, a member of a Class, which is a general thing: John X. Smith is an Instance of the Class Person

- **Distinguished relations**: subclass/isa, instance_of, part_of (part-whole), composition_of, etc.
  - The semantics of these are defined in the meta-level or the upper ontology

- **Class as unary relation**: Person(X)

- **Attribute as relation, reification of relations (as first class citizens, etc.)**

- **Domain & range of relation**
  - works_at(Person, Org)  
    - Domain: Person  
    - Range: Org

- **Slots & roles**: relations “attached” to an instance
  - Slots: in frame systems
  - Roles: in description logics

- **Others**: times, events, processes, purposes, contexts, agents, functions
How To Create a Better Taxonomic Backbone to an Ontology*

• Formal Ontological Analysis: consider “meta” properties such as identity, rigidity, unity (whole)

  • **Identity:** how does an entity change but keep its identity?
    – What are its essential properties?
    – If you change its parts, does it keep its identity?
    – Different properties/same parts, different parts/same properties
    – Persistence over time

  • **Rigidity:** if having a certain property is essential for all instances
    – Having a brain is essential for a person
    – Having an arm is not essential for a person
    – Necessary and sufficient properties
    – Only rigid properties can provide identity

  • **Unity:** parts, whole, connectedness of parts, boundaries of the whole
    – Mereotopology: Parts + Connectedness
    – Collections: the sum is not a whole (five cups of coffee)
    – Plural Wholes: the sum is also a whole (ballplayers vs. team)
    – Statue of Venus vs. the clay that constitutes the statue
      – Venus de Milo: the missing arms were part of the statue of Venus
      – The missing clay was part of the glob of clay that had been formed into the arms

Ontology Modeling Issues: Ontological Levels*, Multiple Dimensions

- Physical
  - Atomic \( (a \ minimal \ grain \ of \ matter) \)
  - Static \( (a \ configuration, \ a \ situation) \)
  - Mereological \( (an \ amount \ of \ matter, \ a \ collection) \)
  - Topological \( (a \ piece \ of \ matter) \)
  - Morphological \( (a \ cubic \ block, \ a \ constellation) \)
- Functional \( (an \ artifact, \ a \ biological \ organ) \)
- Biological \( (a \ human \ body) \)
- Intentional \( (a \ person, \ a \ robot) \)
- Social \( (a \ company) \)

Ontology Modeling Issues: Well-Founded Ontologies - Some Basic Design Principles*

• **Be clear about the domain**
  – particulars (individuals)
  – universals (classes and relations)
  – linguistic entities (nouns, verbs, adjectives...)

• **Take identity seriously**
  – Different identity criteria imply *disjoint classes*

• **Isolate a basic taxonomic structure**
  – Every entity must instantiate a rigid property with identity
  – Physical objects can change parts and remain the same, but amounts of matter cannot
  – Only *sortals* like “person” (as opposite to “red”) are good candidates for being *taxons* (classes in subclass relation)
  – *Sortals*: objects which carry *identity*
  – *Categories*: objects which generalize *sortals*

• **Make an explicit distinction between types and roles**
  (*and other property kinds*)

Ontology Modeling Issues: Reifying Relations?

**Relations**
- Works_At
- Works_On_Craft
- Works_On_Wood

**Ontology**
- Entities
  - Person
  - Occupation
  - Skilled_Labor
  - Carpenter

**Range**
- Occupation

**Domain**
- Person

**VS. Local Attributes:**
- Attribute
- Value
- Carpenter

Copyright © Leo Obrst, MITRE, 2002-09
Ontology Modeling Issues: Guidelines for Building Ontologies*

• How and when to create classes in an ontology that will be useful for reasoning:
  – Every slot (property, relation) on a class must apply to all instances of all subclasses
  – Classes should not be defined solely to allow inheritance of some common attribute by a small number of subclasses
  – Man-made artifacts will be defined primarily by their function and only secondarily by physical attributes

*From a document prepared by Pat Cassidy & other of my ontologist ex-employees, and me, 2000.
Ontology Modeling Issues: Guidelines for Building Ontologies*

Subclass relation

- A subclass must inherit all slots (properties, relations) from its parent and remoter ancestor classes.
- Everything that is true of the instances of a parent class must also be true of instances of the descendent classes (children, etc.).
- Specifically, all slot values and value types of a parent must be true of the slot values and types of the subclasses.
  - e.g. if the class "knife" is a subclass of "CuttingDevice", and a cutting device is defined as a device designed for cutting, then all the members of the subclasses of knife must also be designed for cutting. A steak knife, a bread knife, and a pocket knife are all designed for cutting, and the classes "steak_knife", "bread_knife" and "pocket_knife" are therefore legitimate subclasses of "knife". A class "knife_box" would *not* be a subclass of knife, nor would "knife_handle". There may be doubtful cases, e.g. a butter knife which has a dull blade, but even this is intended for cutting butter (a dictionary definition is: “a small knife with a dull blade, for cutting, serving, or spreading butter.”). A butter knife would thus also qualify as a spreading instrument.
Ontology Modeling Issues: Guidelines for Building Ontologies*

Subclass relation

- There can be a use for a mechanism that will allow "cancellation" of inheritance of a slot/property/relation (i.e., to contradict some assertion that is made about all of the instances of a class)
- Convenient to allow some mechanism to recognize abnormalities about specific instances of things
- For base ontology, don’t need these
When to define classes in order to inherit slots (properties, relations):

- Each slot that we attach to a class asserts something about the object that are members of that class.
- The more we can say about members of a class, the more detailed and accurate our reasoning can be.
- There are two ways of associating slots (attributes) to a class:
  - by making it a subclass of another class
  - by directly attaching slots to the class
  - (Sometimes it is not obvious which way is best)
Ontology Development Methodology: An Example

Approach for Developing Ontologies

1. Identify Domain Expert
2. Identify Users
3. Define Scope (includes identifying verticals which are within scope and enumerating information requirements)
4. Develop Domain Upper Ontology (building off of the buyers' guides)
5. Model Upper Domain Ontology in OntologyBuilder (includes OntologyBuilder training)
6. Identify Appropriate Domain Stds, Ontologies & Reps (and position ourselves on these standard bodies)
7. Evaluate Identified Stds and Reps (against identified scope and information requirements, using the guidelines in the "Reusing Ontologies" paper)
8. Identify and Contract Needed Non-Resident Expertise
9. Conceptualize the Ontology (e.g., develop a glossary, group terms, etc.)
10. Incorporate Approp. Stds, Reps, etc. into Domain Description (also based on e-commerce center taxonomies)
11. Model the Lower Portions of the Ontology in OntologyBuilder
12. Integrate the Ontology with UNSPSC
13. Integrate With other Domain Ontologies
14. Validate the Ontology (for completeness and correctness)
15. Integrate With Upper Ontology

Key:
DDK - Director of Domain Knowledge
DE - Domain Expert
O - Ontologist
SO - Senior Ontologist
DBE - Domain Business Expert
DC - Domain Contractor

* Note: All steps may not necessarily be sequential.
Ontology Development Process Plan: Based on Methontology

*Based on Methontology, Asunción Gómez- Pérez
Ontology Development Process Plan

Build Glossary of Terms (including sources of knowledge)

Identify Concept Classification Tree

Identify Class & Instance Attributes, Values

Identify Concept Relations

Identify Concept Axioms

Identify Attribute Classification Tree

Identify Constants Classification Tree

Identify Constraints, Rules

Identify & Label Privileged Relations (subclass-of, mutually disjoint subclass-of, exhaustive subclass-of), synonyms, acronyms

Identify Value Type, Value Kind (class, instance)

Unit of Measure (and possibly conversion formulae), Precision, Range of Values, Default Value, Cardinality, Description, Source, Time, Author

Copyright © Leo Obrst, MITRE, 2002-09
Agenda, Part 4:
Semantic Web
The Semantic Web

• Current Web is a collection of links and resources: machine-readable, not machine-understandable, semantically-interpretable

• The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation.

• Languages to support machine-interpretable semantics of Web data, artifacts

• Machines will be able to consume machine-readable information, better enabling computers and people to work, learn and exchange knowledge more effectively
Semantic Web Stack

Grid Computing & Grid Services

Intelligent Domain Services, Applications

Use, Intent
Trust
Reasoning/Proof
Richer Semantics
Semantics
Structure
Syntax: Data

Agents, Brokers, Policies

Pragmatic Web
Security/Identity
Inference Engine
OWL (ontologies)
RDF/RDF Schema
XML Schema
XML

• Semantic Brokers
• Intelligent Agents
• Advanced Applications

Semantic Web Technologies

Mature Web Technologies

• Grid & Semantic Grid
Semantic Web: Another View

- Anyone, anywhere can add to an evolving, decentralized “global database”
- Explicit semantics enable looser coupling, flexible composition of services and data

Syntax, Transmission
- “Digital Dial Tone”, Global Addressing
  - HTTP, Unicode, URIs

Structure
- XML Schema

Expose Data & Service Semantics
- RDF/RDF Schema

Enable Reasoning: Proof, Logic
- SWRL, RIF, FOL, Inference

Add Full Ontology Language so Machines can Interpret the Semantics
- OWL

Copyright © Leo Obrst, MITRE, 2002-09
# Semantic Web Services Stack

<table>
<thead>
<tr>
<th>Semantics</th>
<th>Pragmatics</th>
</tr>
</thead>
<tbody>
<tr>
<td>OWL, OWL-S, SAWSDL, SWRL, RIF</td>
<td>Service Entities, Relations, Rules</td>
</tr>
<tr>
<td>RDF/S</td>
<td>Service Instances</td>
</tr>
<tr>
<td>BPEL4WS (Business Process Execution Language for Web Services)</td>
<td>Service Flow &amp; Composition</td>
</tr>
<tr>
<td>Trading Partner Agreement</td>
<td>Service Agreement</td>
</tr>
<tr>
<td>UDDI/WS Inspection</td>
<td>Service Discovery (focused &amp; unfocused)</td>
</tr>
<tr>
<td>UDDI</td>
<td>Service Publication</td>
</tr>
<tr>
<td>WSDL</td>
<td>Service Description</td>
</tr>
<tr>
<td>WS Security</td>
<td>Secure Messaging</td>
</tr>
<tr>
<td>SOAP, REST, etc.</td>
<td>XML Messaging</td>
</tr>
<tr>
<td>HTTP, FTP, SMTP, MQ, RMI over IIOP</td>
<td>Transport</td>
</tr>
</tbody>
</table>

Semantic Web Languages

• Numerous efforts have led to recent convergence on W3C recommendations

• 10 Feb ’04 W3C released recommendations on
  – Resource Description Framework (RDF)
    • Used to represent information and to exchange knowledge in the Web
  – OWL Web Ontology Language (OWL) as W3C
    • Used to publish and share sets of terms called ontologies, supporting advanced Web search, software agents and knowledge management
  – See http://www.w3.org/ for more information

• RDF and OWL are now international standards

• Both RDF and OWL observe the Open World Assumption: new knowledge can always be added to what already exists
What the Languages Provide: RDF/S

- RDFS enables you to make simple, generic statements about your Web object classes, properties
- RDF enables you to make specific statements about your Web object instances (of those classes, properties)
- RDF/S enables you also to make statements about statements (reification), but tells you nothing about those embedded statements
- A set of RDF statements can be viewed in 3 ways:
  - A set of triples: consider them as rows/tuples in a database
  - A directed graph: consider them as a complex, navigatable data structure
  - An inference closure over the relations of the graph: consider them as a machine-interpretable representation of knowledge from which an inference engine can infer new knowledge not expressly encoded

RDF/S, a spectrum of views: database row, graph structured object, inference closure
Resource Description Framework/Schema (RDF/S)

• There is one Language, two levels: RDF is the Language
  – RDFS expresses Class level relations describing acceptable instance level relations
  – RDF expresses Instance level semantic relations phrased in terms of a triple:
  – Statement: <resource, property, value>, <subject, verb, object>, <object1, relation1, object2>

• Resources
  – All things being described by RDF expressions are called resources
    • An entire Web page such as the HTML document
    • Part of a Web page
    • A collection of pages
    • An object that is not directly accessible via the Web
  – Always named by URIs plus optional anchor ids

• Properties
  – A specific aspect, characteristic, attribute, or relation used to describe a resource
  – Specific meaning
  – Permitted values
  – Relationship with other properties

• Statements
  – A specific resource together with a named property plus the value of that property for that resource is an RDF statement

Positive, Existential subset of First Order Logic: no NOT, no ALL:
Can’t represent “John is NOT a terrorist”, “All IBMers are overpaid”
RDF/S Model: Statements

- **Statements**
  - A specific resource together with a named property plus the value of that property for that resource is an RDF statement
  
  - I.e., Triples:
    1. `<Subject Predicate Object>`
    2. `<Resource Property PropertyValue>`
    3. `<Leo,hasColleague,Barry>`

  - PropertyValue can be:
    - another resource (referenced via URI)
    - A literal (primitive datatype defined by XML), i.e., a resource (specified by a URI) or a simple string or other primitive datatype defined by XML
RDF/S Model: A Directed Graph

*“The creator of page http://www.FOOBAR.org/index.html is http://www.FOOBAR.org/staffid/12345”*

http://www.FOOBAR.org/index.html \(\text{subject}\)

\(\text{predicate}\) http://purl.org/dc/elements/1.1/creator

http://www.FOOBAR.org/staffid/12345 \(\text{object}\)

This is also a conceptual graph (with URIs as names)
RDF/S Model: A Directed Graph

Colonel Mustard killed Professor Plum in the Library with the Lead Pipe

http://www.clueless.org/person/#colonel_mustard
http://www.murderInc.com/hit/#kill
http://www.upperOnt.org/#location
http://www.upperOnt.org/#instrument
http://www.murderInc.com/hit/#victim
http://www.clueless.org/person/#professor_plum

http://www.clueless.org/room/#library
http://www.clueless.org/weapon/#lead_pipe

NOTE: This is also a conceptual graph (with URIs as “names”)

Reification: A statement about a statement (but uninterpreted, no truth asserted):
John thinks X, where X = “Colonel Mustard killed Professor Plum in the Library with the Lead Pipe”; don’t know what X ‘means’
What the Languages Provide: OWL

- OWL enables you to make complex, generic statements about your Web object classes, properties
- OWL’s instances are expressed as RDF statements
- OWL has 3 dialects/layers, increasingly more complex: OWL-Lite, OWL-DL, OWL-Full
- OWL is only an ONTOLOGY language (like RDFS) & a Description Logic (classification via subsumption)
- OWL uses everything below it in the Semantic Web stack:
  - Has a presentation/exchange XML syntax, XML datatypes
  - RDF instances
  - RDFS generic (ontology) statements: how depends on the OWL dialect
  - OWL is expressed in an XML exchange and presentation syntax
- OWL enables you to map among ontologies:
  - Import one ontology into another: all things that are true in the imported ontology will thereby be true in the importing ontology
  - Assert that a class, property, or instance in one ontology/knowledge base is equivalent to one in another ontology
# OWL Language Levels*

<table>
<thead>
<tr>
<th>Language Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OWL Full</td>
<td>The complete OWL. For example, a class can be considered both as a collection of instances (individuals) and an instance (individual) itself.</td>
</tr>
<tr>
<td>OWL DL (description logic)</td>
<td>Slightly constrained OWL. Properties cannot be individuals, for example. More expressive cardinality constraints.</td>
</tr>
<tr>
<td>OWL Lite</td>
<td>A simpler language but one that is more expressive than RDF/S. Simple cardinality constraints only (0 or 1).</td>
</tr>
</tbody>
</table>

*Daconta, Obrst, Smith, 2003; cf. also OWL docs at http://www.w3.org/2001/sw/WebOnt/
OWL Lite enables you to define an ontology of classes and properties and the instances (individuals) of those classes and properties.

This and all OWL levels use the `rdfs:subClassOf` relation to defined classes that are subclasses of other classes and which thus inherit those parent classes properties, forming a subsumption hierarchy, with multiple parents allowed for child classes.

Properties can be defined using the `owl:objectProperty` (for asserting relations between elements of distinct classes) or `owl:datatypeProperty` (for asserting relations between class elements and XML datatypes), `owl:subproperty`, `owl:domain`, and `owl:range` constructs.

*Daconta, Obrst, Smith, 2003; cf. also OWL docs at http://www.w3.org/2001/sw/WebOnt/
• **OWL DL extends OWL Lite** by permitting cardinality restrictions that are not limited to 0 or 1

• Also, you can define classes based on specific property values using the `hasValue` construct

• At the OWL DL level, you can create *class expressions* using Boolean combinators (set operators) such as `unionOf`, `intersectionOf`, and `complementOf`

• Furthermore, classes can be enumerated (listed) using the `oneOf` construct or specified to be disjoint using `disjointWith` construct

*Daconta, Obrst, Smith, 2003; cf. also OWL docs at http://www.w3.org/2001/sw/WebOnt/*
OWL Full extends OWL DL by permitting classes to be treated simultaneously as both collections and individuals (instances).

Also, a given `datatypeProperty` can be specified as being `inverseFunctional`, thus enabling, for example, the specification of a string as a unique key.

**Clyde is an elephant.**
Elephant is a species.
Therefore, Clyde is a species.

WRONG!

Clyde is an elephant.
Elephant is a mammal.
Therefore, Clyde is a mammal.

RIGHT!

---

*Daconta, Obrst, Smith, 2003; cf. also OWL docs at http://www.w3.org/2001/sw/WebOnt/  
Human Resource Model in UML
Human Resource Ontology in Protégé
OWL Human Resource Ontology Fragment

- Define a class called `Management_Employee` (1), then a subclass of that class, called `Manager` (2), and finally, an instance of the Manager class – `JohnSmith` (3)
  - The subclass relation is transitive, meaning that inheritance of properties from the parent to the child (subclass of parent) is enabled
  - So a `Manager` inherits all the properties defined for its superclass `Management_Employee`

1. `<owl:Class rdf:ID="Management_Employee"/>`
2. `<owl:Class rdf:ID="Manager"/>`
   
   `<rdfs:subClassOf`
   
   `<rdfs:resource="#Management_Employee"/>`

3. `<Manager rdf:ID="JohnSmith" />`

- Define the property `employs` with domain `Organization` and range, `Employee`

  `<owl:ObjectProperty rdf:ID="employs"/>`
  
  `<rdfs:domain rdf:resource="#Organization"/>`
  
  `<rdfs:range rdf:resource="#Employee"/>`

  `<owl:ObjectProperty>`
OWL Human Resource Ontology Fragment

- Define property `employee_of` with domain `Employee`, range `Organization`
  
  ```xml
  <owl:ObjectProperty rdf:ID="employee_of">
    <rdfs:domain rdf:resource="#Employee"/>
    <rdfs:range rdf:resource="#Organization"/>
  </owl:ObjectProperty>
  ```

- `employee` and `employee_of` are inverses of each other
- In OWL, this inverse relation can be stated in a different way, with the same semantics

  ```xml
  <owl:ObjectProperty rdf:ID="employee_of">
    <owl:inverseOf rdf:resource="#employs" />
  </owl:ObjectProperty>
  ```
OWL Wine Ontology: Snippets*

• **Header, Namespace information**

• **Three Root Classes**
  <owl:Class rdf:ID="Winery"/>
  <owl:Class rdf:ID="Region"/>
  <owl:Class rdf:ID="ConsumableThing"/>

• **Define a Subclass**
  <owl:Class rdf:ID="PotableLiquid"> <rdfs:subClassOf rdf:resource="#ConsumableThing" /> ... </owl:Class>

• **Define an Individual (Instance)**
  <owl:Thing rdf:ID="CentralCoastRegion" /> <owl:Thing rdf:about="#CentralCoastRegion"> <rdf:type rdf:resource="#Region"/> </owl:Thing>

• **Define a property**

Protégé Example: http://protege.stanford.edu/

Class hierarchy: for example, columnists, editors, reporters, and news services are authors

Slot descriptions: for example, editors have names, phone numbers, salaries; they are also responsible for other employees, and contents of sections
Protégé: OWL Pizza Ontology
Protégé: OWLViz
partial view of Pizza
OWL 2 (1)

• OWL 2 is a Proposed W3C Recommendation (22 Sept 2009)*
• Compatible with OWL 1 (04 Feb 2004)
• New features
  – Increased datatype coverage: Designed to take advantage of the new datatypes and clearer explanations available in XSD 1.1 (not yet a recommendation)
  – Syntactic Sugar for more easily saying things in OWL:
  – New constructs that increase expressivity
  – Simple meta-modeling capabilities
  – Extended annotation capabilities
  – Profiles

* http://www.w3.org/TR/2009/PR-owl2-new-features-20090922/
OWL 2 (2)

• Syntactic Sugar for more easily saying things in OWL:
  – DisjointUnion:
    • DisjointUnion(:CarDoor :FrontDoor :RearDoor :TrunkDoor) : A :CarDoor is exclusively either a :FrontDoor, a :RearDoor or a :TrunkDoor and not more than one of them.
  – DisjointClasses
    • DisjointClasses( :LeftLung :RightLung ) : Nothing can be both a :LeftLung and a :RightLung.
  – NegativeObject(Data)PropertyAssertion
    • NegativeObjectPropertyAssertion( :livesIn :ThisPatient :IleDeFrance ) : ThisPatient does not live in the :IleDeFrance region.
  – Self-restriction on Properties: “local reflexivity”
    • SubClassOf( :AutoRegulatingProcess ObjectHasSelf( :regulate ) ) : Auto-regulating processes regulate themselves.
  – Property Qualified Cardinality Restrictions: counted cardinality restrictions (Min, Max, Exact)
    • ObjectMaxCardinality( 3 :boundTo :Hydrogen) : Class of objects bound to at most three different :Hydrogen
  – Many others
OWL 2 (3)

• Simple meta-modeling capabilities:
  – Punning: allows different uses of the same term and an individual
  – OWL 2 DL still imposes certain restrictions: it requires that a name cannot be used for both a class and a datatype and that a name can only be used for one kind of property; semantically names are distinct for reasoners

• Annotations:
  – AnnotationAssertion: for annotation of ontology entities
  – Annotation: for annotations of axioms and ontologies
  – Etc.

• New constructs that increase expressivity
  – Declarations: a declaration signals that an entity is part of the vocabulary of an ontology. A declaration also associates an entity category (class, datatype, object property, data property, annotation property, or individual) with the declared entity
  – Declaration( NamedIndividual( :Peter ) ): Peter is declared to be an individual
• Profiles:
  – OWL 1 defined two major dialects, OWL DL and OWL Full, and one syntactic subset (OWL Lite)
  – Needs:
    • Some large-scale applications (e.g., in the life sciences) are mainly concerned with language scalability and reasoning performance problems and are willing to trade off some expressiveness in return for computational guarantees, particularly w.r.t. classification
    • Other applications involve databases and so need to access such data directly via relational queries (e.g., SQL)
    • Other applications are concerned with interoperability of the ontology language with rules and existing rule engines
  – Therefore, 3 profiles (sublanguages, i.e., syntactic subsets of OWL 2) are defined: OWL 2 EL, OWL 2 QL, and OWL 2 RL*

• And more!

* http://www.w3.org/TR/2009/PR-owl2-profiles-20090922/
Semantic Web Rules: RuleML, SWRL (RuleML + OWL), RIF

- **Reaction rules** can be reduced to general rules that return no value. Sometimes these are called “condition-action” rules. Production rules in expert systems are of this type.
- **Transformation rules** can be reduced to general rules whose 'event' trigger is always activated. A Web example of transformation rules are the rules expressed in XSLT to convert one XML representation to another. “Term rewrite rules” are transformation rules, as are ontology-to-ontology mapping rules.
- **Derivation rules** can be reduced to transformation rules that like characteristic functions on success just return true. Syntactic A |-p B and Semantic Consequence A |=p B are derivation rules.
- **Facts** can be reduced to derivation rules that have an empty (hence, 'true') conjunction of premises. In logic programming, for example, facts are the ground or instantiated relations between “object instances”.
- **Queries** can be reduced to derivation rules that have – similar to refutation proofs – an empty (hence, 'false') disjunction of conclusions or – as in 'answer extraction' – a conclusion that captures the derived variable bindings.
- **Integrity constraints** can be reduced to queries that are 'closed' (i.e., produce no variable bindings).

So Which Rules Are Useful, Good, Bad, Ugly?

★ Good
- Logical rules are declarative, confirmable by human beings, machine semantically-interpretable, non-side-effecting
- Logical rules can express everything that production (expert system) rules, procedural rules can
- Logical rules can express business, policy rules, static/dynamic rules

★ Bad
- Rules expressed in procedural code if-then-else case statements are non-declarative, inspectable by human beings, confirmable with documentation and observance of conformance to documentation, side-effecting (ultimate side-effect: negating a value and returning true for that value)

★ Ugly
- Expert systems rules “simulate” inference, are pre-logical, have side-effects, tend toward non-determinism, force all knowledge levels to the same level (this is why ontologies and ontological engineering came about), are horrible to debug
Example: Inference and Proof

Proof Using Inference Rule of Modus Ponens

Given: If motherOf is a subProperty of parentOf, and Mary is the mother of Bill, then Mary is the parentOf Bill

- motherOf is a subProperty of parentOf
- Mary is the motherOf Bill

Infer: Mary is the parentOf Bill

Deduction A method of reasoning by which one infers a conclusion from a set of sentences by employing the axioms and rules of inference for a given logical system.

Rule Interchange Format (RIF)*

- RIF is a rule language based on XML syntax
- RIF provides multiple versions, called dialects:
  - **Core**: the fundamental RIF language, and a common subset of most rule engines (It provides "safe" positive datalog with builtins)
  - **BLD (Basic Logic Dialect)**: adds to Core: logic functions, equality in the then-part, and named arguments (This is positive Horn logic, with equality and builtins)
  - **PRD (Production Rules Dialect)**: adds a notion of forward-chaining rules, where a rule fires and then performs some action, such as adding more information to the store or retracting some information (This is comparable to production rules in expert systems, sometimes called condition-action, event-condition-action, or reaction rules)

http://www.w3.org/2005/rules/wiki/RIF_Working_Group
http://www.w3.org/2005/rules/wiki/RIF_FAQ
Trust

• Trust requires
  – **Identity:** knowing that the person, agent, organization, software application, or Semantic Web ontology is who they say they are; digital signatures, PKI, etc., help establish this
  – **Credibility, Trustworthiness:** knowing that the Semantic Web artifact was created by a reputable agent, organization, i.e., one that has a reputation for quality, truth, response to customers, commitment to error correction, and adherence to self-advertised use and intent policies
  – **Proof:** being able to prove that the response you, your agent, or your inference engine is given to a query, function call, or service request on the Semantic Web is indeed true, and correctly follows; an explanation or trace that ensures this
  – **Security and Privacy:** being able to ensure that access to your property and to the rights you grant are strictly enforced at the sufficient granularity of detail you or your policy requires
Use / Intent

• Semantic Web artifacts define their meaning using ontologies, fact/knowledge bases, and Semantic Web services
• Those semantic models and services are intended to
  – Represent what you mean
  – Be used by others in the way you meant them to be used
• The *Pragmatic Web* concerns the correct interpretation of semantic models and services in context
  – i.e., according to the use and intent they were created for, perhaps in a specific process/workflow model
  – By a human, an agent, or another Semantic Web service
• Policy: in many cases, you will declare a Semantic Web policy about how your Semantic Web models and services need to be interpreted and used
  – Like business rules and pragmas in computer programming
  – Coercions will be needed, but violations should be flagged – as violating the use and intent of your semantics
  – Policy helps stabilize the Semantic Web
  – Policy helps maintain your and your site’s credibility
  – Policy helps agents and services interpret how they should interpret your models and services
Where is the Technology Going?

• “The Semantic Web is very exciting, and now just starting off in the same grassroots mode as the Web did 10 years ago ... In 10 years it will in turn have revolutionized the way we do business, collaborate and learn.”
  – Tim Berners-Lee, CNET.com interview, 2001-12-12

• We can look forward to:
  – Semantic Integration/Interoperability, not just data interoperability
  – Applications and services with trans-community semantics
  – Device interoperability in the ubiquitous computing future: achieved through semantics & contextual awareness
  – True realization of intelligent agent interoperability
  – Intelligent semantic information retrieval & search engines
  – Next generation semantic electronic commerce/business & web services
  – Semantics beginning to be used once again in NLP

Key to all of this is effective & efficient use of explicitly represented semantics (ontologies)
The Point (s)

• The point is that we need to model our best human theories (naïve or scientific, depending on our system needs)
• In a declarative fashion (so that humans can easily verify them)
• And get our machines to work off them, as models of what humans do and mean
• We need to build our systems, our databases, our intelligent agents, and our documents on these models of human meaning
• These models must:
  – Represent once (if possible)
  – Be semantically reasonable (sound)
  – Be modular (theories or micro-theories or micro-micro-theories)
  – Be reused. Be composable. Be plug-and-playable
  – Be easily created and refined. Adaptable to new requirements, dynamically modifiable
  – Be consistent or boundably consistent so that our machines can reason and give use conclusions that are sound, trustable or provable, and secure
• We need to enable machines to come up to our human conceptual level (rather than forcing humans to go down to the machine level)
Conclusion

• We have discussed Syntax and Semantics, and what the distinctions are
• Ontology Spectrum and the Range of Semantic Models: from Taxonomy (both Weak and Strong) to Thesaurus to Conceptual Model (Weak Ontology) to Logical Theory (Strong Ontology)
• Knowledge Representation: Semantic Networks to Frame-based KR to Description Logics to Full Logic (Propositional and FOL), including Logic Programming
• Ontology Engineering: How to Model, i.e., Concepts and Relationships, Principles
• Semantic Web: RDF/S, OWL, SWRL, RIF, more: trust
What do we want the future to be?

- 2100 A.D: models, models, models
- There are no human-programmed programming languages
- There are only Models

Ontological Models

Knowledge Models

Belief Models

Application Models

Presentation Models

Target Platform Models

Executable Code

Transformations, Compilations
Conclusions: Some Philosophers and Ontology

• Aristotle: “To be is to be”
• Nietzsche: “To do is to be”
• Sartre: “To be is to do”
• Husserl: “To do should be to be”
• Sinatra: “Shoo be do be do do”
  – My way or the highway?
Thank You!

Questions? lobrst@mitre.org