

Introduction to Ontologies

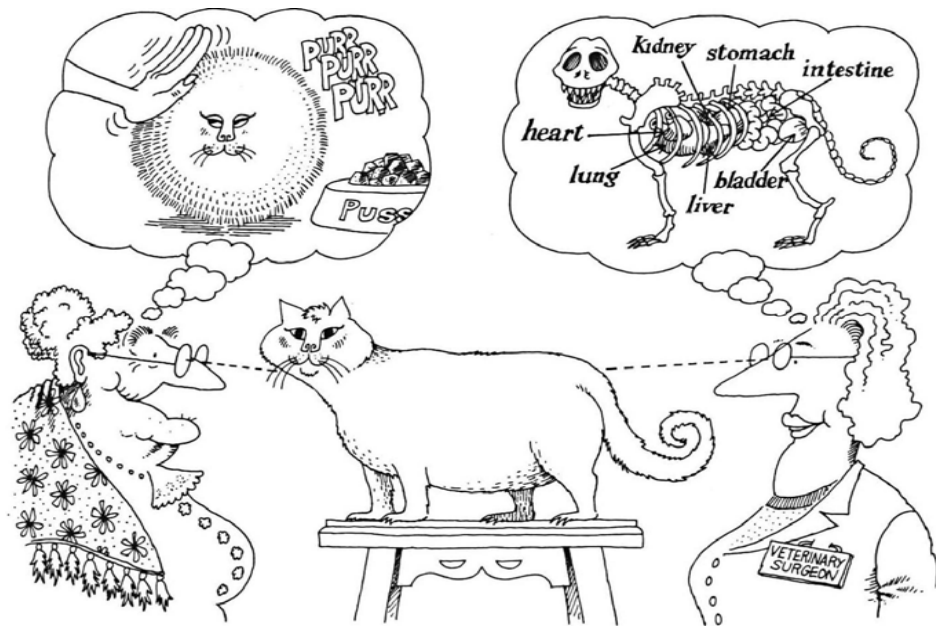
Summer School

Applications of 3D Shapes: Ontologies, Software Tools and
Industrial Case Studies

July 19-25, 2006, Tallinn, Estonia

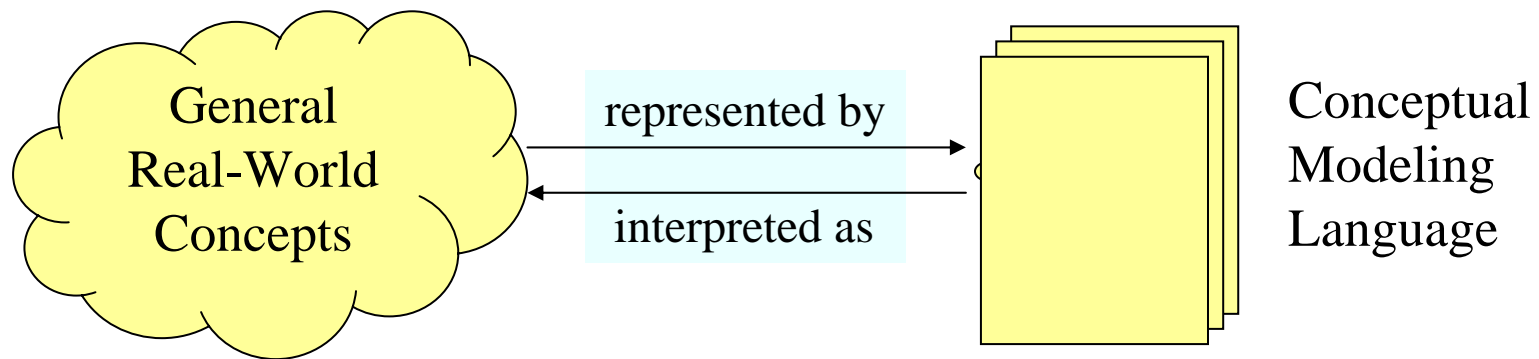
Marios Pitikakis (ITI)

- Information modeling means “models of information”
(NOT “models made by information”)
- We are interested in models of information about the real world, or *somebody's conception* of the real world.



- **Goal:** improve models and tools for representing information and processes
- **Why:** narrow the gap between concepts in the real world and their representation in conceptual models
- **How:** identify and formalize modeling primitives, like generic relationships, for more accurate and intuitive descriptions of real-world concepts

- “Conceptual Modeling is the activity of formally describing some aspects of physical and social world around us for the purposes of understanding and communication” (*Mylopoulos*)
- “Conceptual models offer abstract views on certain aspects of the real-world (descriptive view)” (*Yourdon*)



Basic building blocks of conceptual models and structuring mechanisms:

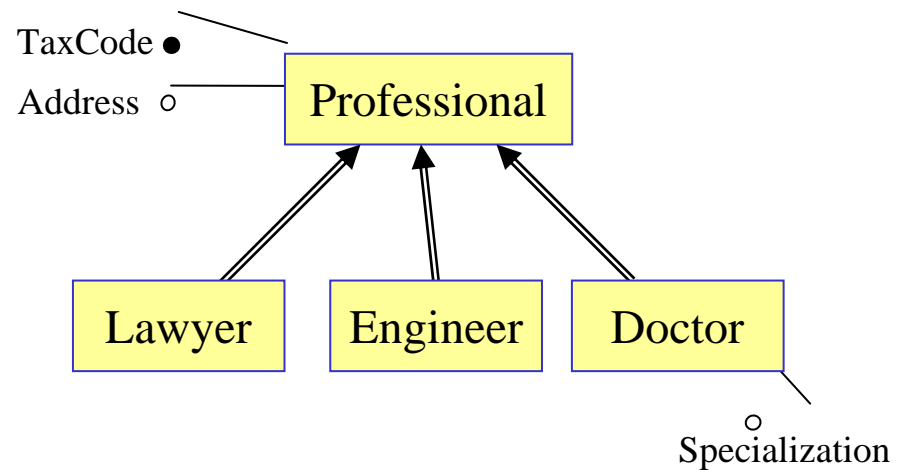
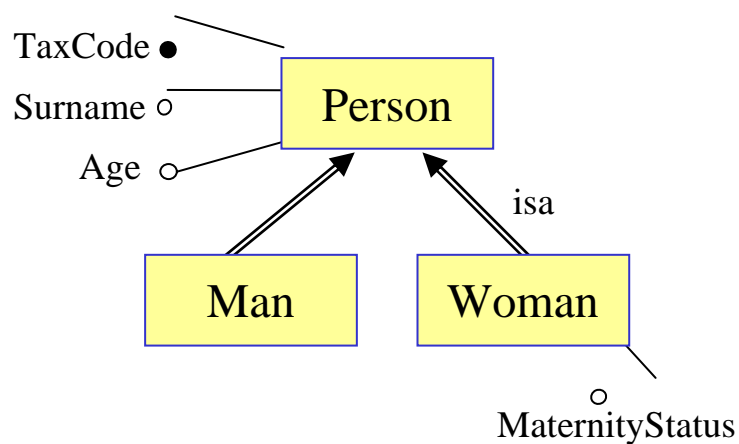
- **Primitive Terms** - concepts built into a conceptual model
(e.g. Entity, Activity, Goal, Time, Space,...)
- **Abstraction Mechanisms** (also called **Semantic Relationships**) -- primitive mechanisms for structuring
(e.g. **Generalization, Aggregation, Classification, Materialization,...**)

- Entities represent classes of objects that have common properties and autonomous existence.
 - *City, Department, Employee, Purchase and Sale* are examples of entities for a commercial organization.
- An instance of an entity is an object in the class represented by the entity.
 - *Stockholm, Helsinki, Tallinn*, are examples of instances of the entity *City*, and the employees *Peterson* and *Johanson* are examples of instances of the *Employee* entity.

- Relationships represent logical links between two or more entities.
 - *Residence* is an example of a relationship that can exist between the entities *City* and *Employee*; *Exam* is an example of a relationship that can exist between the entities *Student* and *Course*.
- An instance of a relationship is an n-tuple made up of instances of entities, one for each of the entities involved.
 - The pair (Johanssen, Stockholm), or the pair (Peterson, Oslo), are examples of instances in the relationship *Residence*.

- Attributes describe the elementary properties of entities or relationships.
 - For example, *Surname*, *Salary* and *Age* are possible attributes of the *Employee* entity, while *Date* and *Mark* are possible attributes for the relationship *Exam* between *Student* and *Course*.
- An attribute associates each instance of an entity (or relationship) with a value belonging to a set known as the **domain** of the attribute.
- The domain contains the admissible values for the attribute.

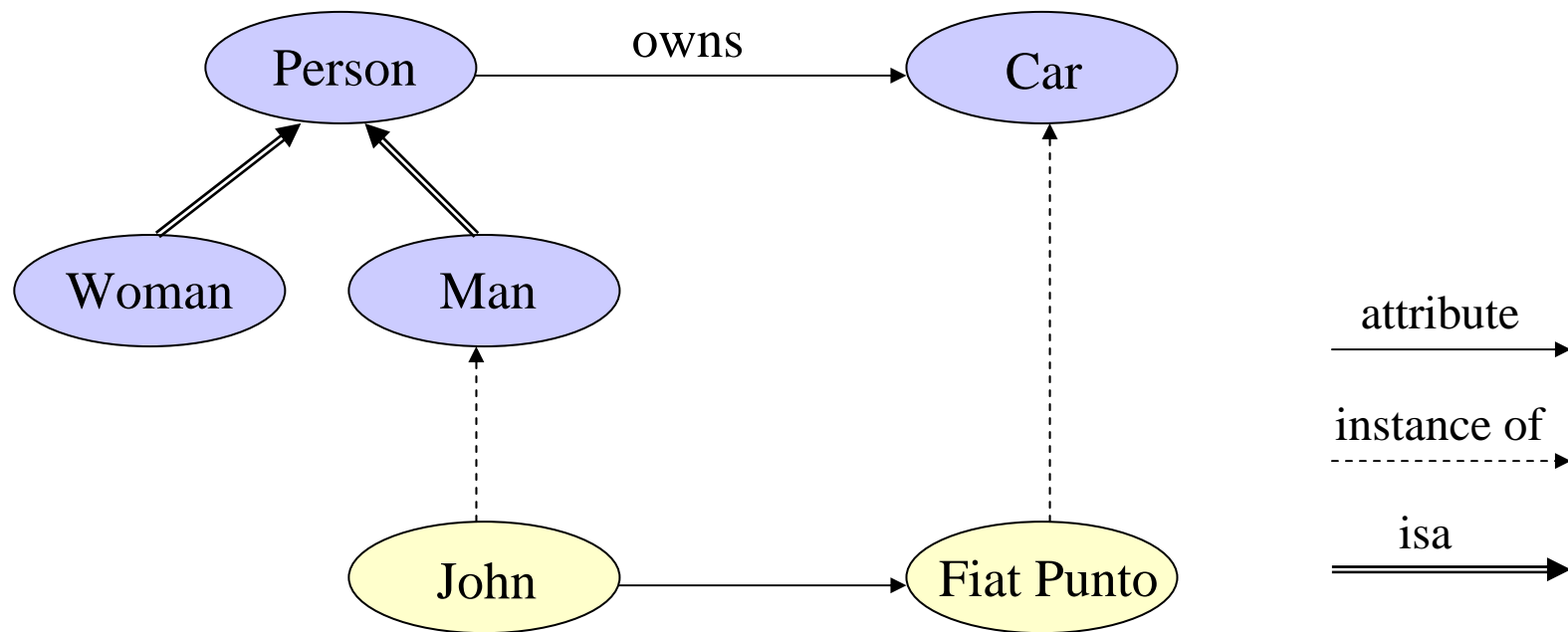
- Generalizations represent logical links between an entity E, known as **parent** entity, and one or more entities E1, ..., En called **child** entities, of which E is more general, in the sense that they are a particular case.
- In this situation we say that E is a **generalization** of E1, ..., En and that the entities E1, ..., En are **specializations** (noted isa) of E.



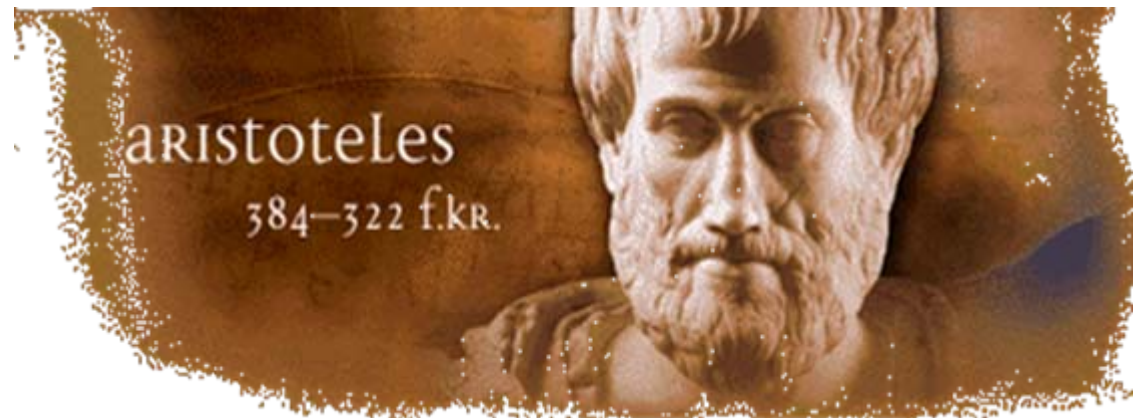
- Every instance of a child entity is also an instance of the parent entity.
- Every property of the parent entity (attribute, relationship or other generalization) is also a property of a child entity. This property of generalizations is known as ***inheritance***.

Objects are structured along three main hierarchies:

- the **classification** hierarchy
- the **generalization/specialization** hierarchy
- the **aggregation (attribute)** hierarchy



- A discipline of Philosophy
 - Meta-physics dates back to Aristotle
 - Ontology dates back to 17th century
- The science of what is ("being qua being")
- Ontology derives from the Greek word "*on*" (being) and "*logos*" (*word, speech, or reason*).



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

- 'Conceptualization' refers to an abstract model of phenomena in the world by having identified the relevant concepts of those phenomena.
- 'Explicit' means that the type of concepts used and their constraints are explicitly defined.
- 'Formal' refers to the fact that the ontology should be machine readable.
- 'Shared' reflects that ontology should capture consensual knowledge accepted by the communities. Ontologies are used to facilitate knowledge sharing.

- A **taxonomy** – also from Greek "*taxis*" (arrangement, order, division) and "*nomos*" (law) - is a single hierarchical structure of objects.
- An **ontology** is more than a taxonomy or classification of terms. Ontologies include richer relationships between terms. It is these rich relationships that enable the expression of domain-specific knowledge. This is a key distinction.

Five kinds of components:

(1) classes:

- concepts of the domain or tasks, which are usually organized in taxonomies

(2) relations:

- a type of interaction between concepts of the domain (e.g. subclass-of, is-a)

(3) instances:

- to represent specific elements (e.g. Student called Peter is the instance of Student class)

(4) functions:

- a special case of relations in which the n -th element of the relationship is unique for the $n-1$ preceding elements (e.g. Price-of-a-used-car can define the calculation of the price of the second-hand car on the car-model, manufacturing data and kilometers)

(5) axioms:

- model sentences that are always true (e.g. if the student attends both A and B course, then he or she must be a second year student)

- Knowledge Representation ontologies
 - capture the representation primitives used to formalize knowledge in KR paradigm
- General/Common ontologies
 - vocabulary related to things, events, time, space, etc.
- Meta-ontologies
 - reusable across domains
- Domain ontologies
 - vocabularies about the concepts in a domain

- Task ontologies
 - a systematic vocabulary of the terms used to solve problems associated with tasks that may or may not come from the same domain
- Domain-task ontology
 - task ontology reusable in a given domain
- Application ontology
 - necessary knowledge for modeling a particular domain

- The reason ontologies are becoming popular is largely due to what they promise: *a shared and common understanding of a domain that can be communicated between people and application systems.*
- Semantic Interoperability
 - Generalized database integration
 - Virtual Enterprises
 - e-commerce
 - e-Science
- Information Retrieval
 - Decoupling user vocabulary from data vocabulary
 - Query answering
 - Natural Language Processing

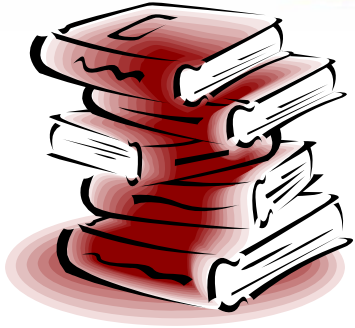
- Fundamentally, ontologies are used to improve communication between either humans or computers. Broadly, these may be grouped into the following three areas:
 - to assist in **communication** between humans. Here, an unambiguous but informal ontology may be sufficient.
 - to achieve **interoperability** among computer systems achieved by translating between different modelling methods, paradigms, languages and software tools. Here, the ontology is used as an interchange format.
 - to improve the process and/or quality of engineering software systems.

Systems Engineering Benefits: In particular,

- Re-Usability: the ontology is the basis for a formal encoding of the important entities, attributes, processes and their inter-relationships in the domain of interest. This formal representation may be a re-usable and/or shared component in a software system.
- Search: an ontology may be used as metadata, serving as an index into a repository of information.
- Reliability: a formal representation also makes possible the automation of consistency checking resulting in more reliable software.

- Specification: the ontology can assist the process of identifying requirements and defining a specification for an IT system (knowledge based, or otherwise).
- Maintenance: use of ontologies in system development, or as part of an end application, can render maintenance easier in a number of ways.
- Knowledge Acquisition: using an existing ontology as the starting point and basis for guiding knowledge acquisition when building knowledge-based systems may increase speed and reliability.

- An ontology language usually introduces **concepts** (classes, entities), **properties** of concepts (slots, attributes, roles), **relationships** between concepts (associations), and additional **constraints**.
- Ontology languages may be simple (e.g., having only concepts and taxonomies), frame-based (having only concepts and properties), or logic-based (e.g. DAML+OIL and OWL).
- Ontology languages are typically expressed by means of diagrams.
- Entity-Relationship schemas and UML class diagrams can be considered as ontologies.



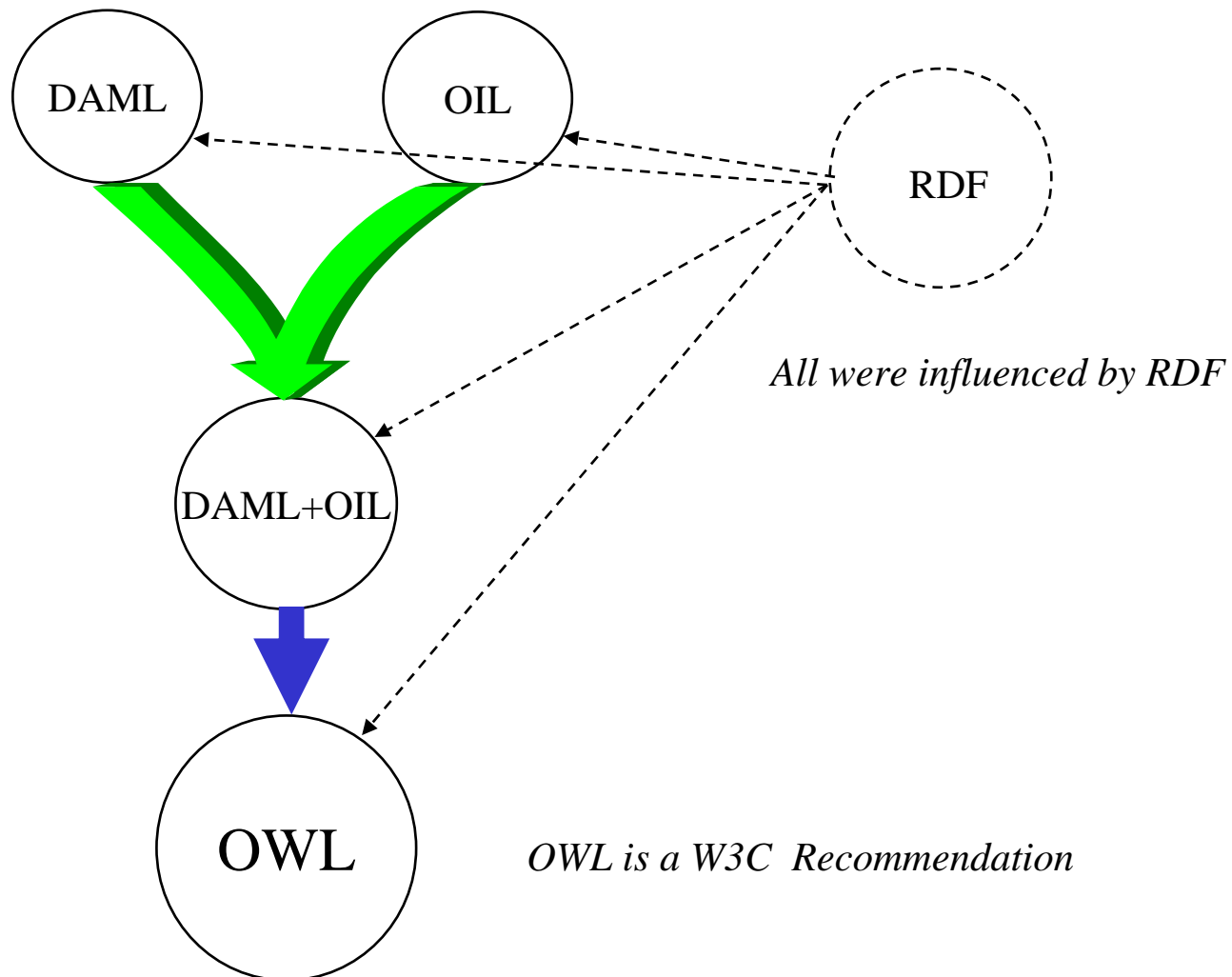
- Gruber, T., "The Role of a Common Ontology in Achieving Sharable, Reusable Knowledge Bases," Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning, Cambridge, 601-602.
- Guarino, N. and Welty, C., "Ontological Analysis of Taxonomic Relations," in A. Länder and V. Storey (eds.), Proceedings of ER-2000: The International Conference on Conceptual Modeling, Springer Verlag LNCS 1920
- Van Heijst, G., Schreiber, A.Th., Wielinga, B.J., "Using explicit ontologies in KBS development", *Int. J. Human-Computer Studies*, Vol. 45, pp. 183-292, 1997.
- Studer, R., Benjamins, V.R., Fensel, D., "Knowledge engineering: Principles and methods", *Data & Knowledge Engineering*, Vol. 53, pp. 161-197, 1998.

- The following paper analyzes the most representative ontology languages created for the Web, and compare them using a common framework
 - “Ontology Languages for the Semantic Web”, IEEE Intelligent Systems, Jan/Feb 2002, pp 54-60
 - **XOL** (XML-based Ontology Exchange Language)
 - **SHOE** (Simple HTML Ontology Extension)
 - **OML** (Ontology Markup Language)
 - **RDF(S)** (Resource Description Framework (Schema))
 - **OIL** (Ontology Interchange Language)
 - **DAML+OIL** (DARPA Agent Markup Language + OIL)
 - **OWL** (Ontology Web Language)

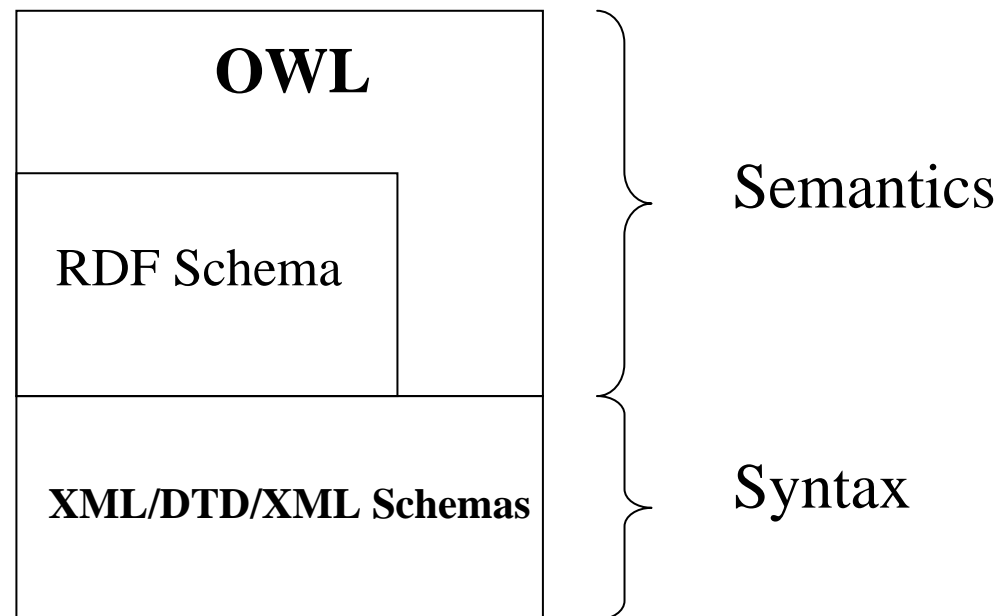
Desirable features identified for a Web Ontology Language :

- **Compatible** with existing Web standards (XML, RDF, RDFS)
- **Easy to understand** and use
- **Formally specified** and of “adequate” expressive power
- Possible to provide **automated reasoning** support

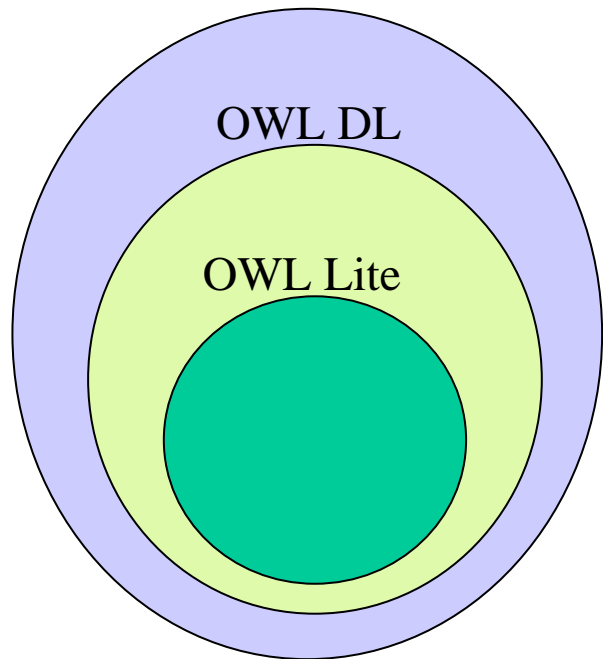
- OWL is a W3C Recommendation (since Feb. 2004)
- The **purpose** of OWL is identical to RDFS i.e. to provide an XML vocabulary to define classes, properties and their relationships.
 - RDFS enables you to express very basic relationships and has limited inferencing capability.
 - OWL enables you to express much richer relationships, thus yielding a much enhanced inferencing capability.
- The **benefit** of OWL is that it facilitates a much greater degree of inferencing than you get with RDF Schemas.



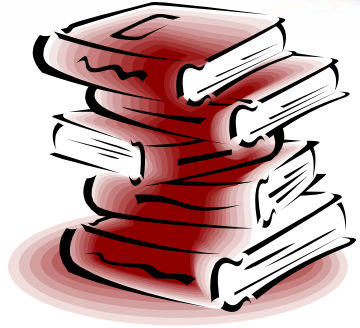
- OWL and RDF Schema enables *machine-processable semantics*



- Depending on the intended usage, OWL provides three increasingly expressive sublanguages



- Full: Very expressive, no computation guarantees
- DL (Description Logic): Maximum expressiveness, computationally complete
- Lite: Simple classification hierarchy with simple constraints.



- W3C Documents
 - Guide
<http://www.w3.org/TR/owl-guide/>
 - Reference
<http://www.w3.org/TR/owl-ref/>
 - Semantics and Abstract Syntax
<http://www.w3.org/TR/owl-semantics/>
- OWL Tutorial
 - <http://www.cs.man.ac.uk/~horrocks/ISWC2003/Tutorial/>

Tool environment addresses three key aspects:

- Acquiring ontologies and linking them with large amounts of data. For reasons of quality this process requires the human in the loop to build and manipulate ontologies using ontology editors.
- Storing and maintaining ontologies and their instances.
- Querying and browsing semantically enriched information sources.

- Commercial Ontology Support Tools
 - SNOBASE
 - Cerebra
- Reasoners
 - FaCT
 - Racer
 - Cerebra
- Editors
 - OWL plug-in for Protégé
- APIs
 - Jena
 - Cerebra
- Parser/Validators

- A large number of open source tools and toolkits have been developed for RDF/S and OWL (parsers, reasoners, APIs, etc.)
- In various languages (Java, Lisp, Python, PHP, Perl, C, C#, etc.)
 - See also <http://www.w3.org/RDF/#developers>
- Personal favorite - Jena2 (Java)
 - <http://jena.sourceforge.net/>

- Jena2 is powerful, flexible and easy Java based framework for building Semantic Web applications. It provides a programmatic environment for [RDF](#), [RDFS](#) and [OWL](#).
- Jena is open source and grown out of work with the [HP Labs Semantic Web Programme](#).
- The Jena Framework includes:
 - A RDF API and RDF/XML Parser
 - Reading and writing RDF in RDF/XML, N3 and N-Triples
 - An OWL API
 - In-memory and persistent storage (MySQL, Oracle and Postgres)
 - [SPARQL](#) query engine
 - A Reasoning Subsystem - Inference capabilities for RDFS and OWL

Ontology development in AIM@SHAPE

Summer School

Applications of 3D Shapes: Ontologies, Software Tools and
Industrial Case Studies

July 19-25, 2006, Tallinn, Estonia

Marios Pitikakis (ITI)

The motivation behind the development of ontologies falls in the following areas:

- Sharing a common understanding of the information in a knowledge domain;
- Improving interoperability among applications that use the domain knowledge;
- Making domain assumptions explicit so that applying changes as these assumptions evolve becomes easier;
- Enabling re-use of the domain knowledge.

Objectives:

- Define common metadata for shape models and shape processing tools.
- Development and evolution of ontologies for the formalization of the various domains of knowledge.
- Build a common conceptual framework to be used by all domain ontologies and the Digital Shape Workbench (DSW).
- Integrate and validate the ontologies and the corresponding metadata.

- Ontology development in the network has been mainly focused on three different areas:
 - Virtual Humans
 - Shape Acquisition and Processing
 - Product Design
- Concepts that are shared by all domain ontologies have lead to the creation of two “common ontologies” (one for shapes and one for tools), as a first attempt for a unified ontology framework.
- The goal is to create higher level ontologies which can be extended by the domain ontologies to express metadata for each domain ontology.