Ontologies: An Introduction

Claudia d'Amato

Co-Author: Nicola Fanizzi

Dipartimento di Informatica • *Università degli Studi di Bari*
Campus Universitario, Via Orabona 4, 70125 Bari, Italy

Bari, November, 7 2007

Contents

The Origin of the "Ontology"

The term **Ontology** has its origin in philosophy

- **Ontology: the branch of philosophy which deals with the nature and the organisation of reality**

In this sense the Ontology tries to answer to the question:

- *What is being?* or, in a meaningful reformulation:
- *What are the features common to all beings?*

"Ontology": recently adopted in several fields of computer science and information science ⇒ several meaning have been assigned to the "Ontology" term

"Ontology" in Computer Science and Information Science

[Guarino et al. 1995]

- Ontology as a specific *"syntactic"* object
  - Ontology as representation of conceptual system via a logical theory
  - Ontology as the vocabulary used by a logical theory
  - Ontology as a meta-level specification of a logical theory
- Ontology as a conceptual *"semantic"* entity
  - Ontology as *informal* conceptual system
  - Ontology as *formal* semantic account
- Ontology as specification of a *conceptualization*
  - an intensional semantic structure which encodes the implicit rules constraining the structure of a piece of reality.

Different kinds of Ontologies

- **Domain Ontology:** models a specific domain. It represents the particular meanings of terms as they apply to that domain (ex. *CHEMICALS, Gene Ontology*). The word *card* has different meaning:
  - An ontology about the domain of *poker* would model the *playing card*
  - An ontology about the domain of *computer hardware* would model the *punch card* and *video card* meanings.
- **Upper Ontology** (or foundation ontology): is a model of the common objects that are generally applicable across a wide range of domain ontologies.
  - It contains a core glossary in whose terms can be used to describe a set of domains. Ex. *Dublin Core, GFO, OpenCyc/ResearchCyc, SUMO, and DOLCE*

Ontology Role and Usage

Def. An *ontology is* a formal conceptualization of a domain that is shared and reused across domains, tasks and group of people **[A. Gomez Perez et al. 1999]**

- The *ontology role* is to make *semantics explicit*. Thus ontologies are *used* for:
  - Constituting a community reference
  - Sharing consistent understanding of what information means
  - Making possible Knowledge Reuse and Sharing
  - Increasing Interoperability between systems

Ontology: Basic elements...

- **Individuals:** are the "ground level" components of an ontology.
  - Individuals can be: **1)** *concrete objects of a domain* i.e. *people, animals, automobiles, molecules...* **2)** *abstract individuals* i.e. *numbers and words*.
- **Concepts:** are collections of objects. They may contain individuals, other classes, or a combination of both. Some examples of classes:
  - *Molecule*, the class of all molecules
  - *Vehicle*, the class of all vehicles
  - *Car*, the class of all cars

...Ontology: Basic elements

- **Attributes:** *describe the objects* in the ontology.
  Ex.: the Ford Explorer object has attributes:
  - Number-of-doors: 4
  - Transmission: 6-speed
- **Relationships:** make explicit the links between objects.
  A relationship can be model as:
  1. an *attribute whose value is another object* in the ontology,
     - **Ex.**: given the objects Ford Explorer and Ford Bronco, the attribute Successor:Ford Explorer of Ford Bronco means that Explorer is the modeled that replaced Bronco.
  2. a mathematical relation
     - **Ex.**: Successor(Ford Bronco,Ford Explorer)
- Much of the power of ontologies comes from the ability to describe these relations.

Ontology Representation Languages

- **Informal:** natural language
- **Semi-Formal:** limited structured form of natural language
- **Formal:** formal language with formal semantics
  - *CycL:* developed in the Cyc project. It is based on first-order predicate calculus with some higher-order extensions.
  - *RIF* (Rule Interchange Format) and *F-Logic:* combine ontologies and rules.
  - **OWL:** developed as a follow-on from RDF and RDFS, and earlier ontology language projects: OIL, DAML, DAML+OIL. *OWL is intended to be used over the World Wide Web*,
    - Supported by *well-founded semantics* of *DLs*
    - together with a series of available automated *reasoning services* allowing to derive logical consequences from an ontology

DL: The Reference Representation Language

Basics elements of DL

- Primitive *concepts* $N_C = \{C, D, \ldots\}$: subsets of a domain
- Primitive *roles* $N_R = \{R, S, \ldots\}$: binary relations on the domain
- *Interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where $\Delta^{\mathcal{I}}$: *domain* of the interpretation and $\cdot^{\mathcal{I}}$: *interpretation function* that assigns to each primitive concept C a subset $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and assigns to each primitive role R a binary relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

Building Complex Concept Descriptions

Name	Syntax	Semantics
atomic negation	$\neg A,\ A \in N_C$	$A^\mathcal{I} \subseteq \Delta^\mathcal{I}$
full negation	$\neg C$	$C^\mathcal{I} \subseteq \Delta^\mathcal{I}$
concept conj.	$C \sqcap D$	$C^\mathcal{I} \cap D^\mathcal{I}$
concept disj.	$C \sqcup D$	$C^\mathcal{I} \cup D^\mathcal{I}$
full exist. restr.	$\exists R.C$	$\{a \in \Delta^\mathcal{I} \mid \exists b\ (a,b) \in R^\mathcal{I} \land b \in C^\mathcal{I}\}$
universal restr.	$\forall R.C$	$\{a \in \Delta^\mathcal{I} \mid \forall b\ (a,b) \in R^\mathcal{I} \rightarrow b \in C^\mathcal{I}\}$
at most restr.	$\leq nR$	$\{a \in \Delta^\mathcal{I} \mid \mid \{b \in \Delta^\mathcal{I} \mid (a,b) \in R^\mathcal{I}\} \mid \leq n$
at least restr.	$\geq nR$	$\{a \in \Delta^\mathcal{I} \mid \mid \{b \in \Delta^\mathcal{I} \mid (a,b) \in R^\mathcal{I}\} \mid \geq n$
qualif. at most r.	$\leq nR.C$	$\{a \in \Delta^\mathcal{I} \mid \mid \{b \in \Delta^\mathcal{I} \mid (a,b) \in R^\mathcal{I} \land b \in C^\mathcal{I}\}$
qualif. at least r.	$\geq nR.C$	$\{a \in \Delta^\mathcal{I} \mid \mid \{b \in \Delta^\mathcal{I} \mid (a,b) \in R^\mathcal{I} \land b \in C^\mathcal{I}\}$
one-of	$\{a_1, a_2, \ldots a_n\}$	$\{a \in \Delta^\mathcal{I} \mid a = a_i, 1 \leq i \leq n\}$
has value	$\exists R.\{a\}$	$\{b \in \Delta^\mathcal{I} \mid (b, a^\mathcal{I}) \in R^\mathcal{I}\}$
inverse of	R^-	$\{(a,b) \in \Delta^\mathcal{I} \times \Delta^\mathcal{I} \mid (b,a) \in R^\mathcal{I}\}$

Knowledge Base & Subsumption

$\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

- *T-box* $\mathcal{T}$ is a set of definitions $C \equiv D$, meaning $C^{\mathcal{I}} = D^{\mathcal{I}}$, where C is the concept name and D is a description
- *A-box* $\mathcal{A}$ contains extensional assertions on concepts and roles e.g. $C(a)$ and $R(a, b)$, meaning, resp., that $a^{\mathcal{I}} \in C^{\mathcal{I}}$ and $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$.

Subsumption

Given two concept descriptions C and D, C *subsumes* D, denoted by $C \sqsupseteq D$, iff for every interpretation $\mathcal{I}$, it holds that $C^{\mathcal{I}} \supseteq D^{\mathcal{I}}$

TBox: Example

Primitive Concepts: $N_C = \{\text{Female}, \text{Male}, \text{Human}\}$.
Primitive Roles:
$N_R = \{\text{HasChild}, \text{HasParent}, \text{HasGrandParent}, \text{HasUncle}\}$.
$\mathcal{T} = \{$ Woman $\equiv$ Human $\sqcap$ Female; Man $\equiv$ Human $\sqcap$ Male
Parent $\equiv$ Human $\sqcap$ $\exists$HasChild.Human
Mother $\equiv$ Woman $\sqcap$ Parent
Father $\equiv$ Man $\sqcap$ Parent
Child $\equiv$ Human $\sqcap$ $\exists$HasParent.Parent
Grandparent $\equiv$ Parent $\sqcap$ $\exists$HasChild.($\exists$ HasChild.Human)
Sibling $\equiv$ Child $\sqcap$ $\exists$HasParent.($\exists$ HasChild $\geq$ 2)
Niece $\equiv$ Human $\sqcap$ $\exists$HasGrandParent.Parent $\sqcup$ $\exists$HasUncle.Uncle
Cousin $\equiv$ Niece $\sqcap$ $\exists$HasUncle.($\exists$ HasChild.Human)$\}$.

ABox: Example

$\mathcal{A}$ = {Woman(Claudia), Woman(Tiziana), Father(Leonardo), Father(Antonio), Father(AntonioB), Mother(Maria), Mother(Giovanna), Child(Valentina), Sibling(Martina), Sibling(Vito), HasParent(Claudia,Giovanna), HasParent(Leonardo,AntonioB), HasParent(Martina,Maria), HasParent(Giovanna,Antonio), HasParent(Vito,AntonioB), HasParent(Tiziana,Giovanna), HasParent(Tiziana,Leonardo), HasParent(Valentina,Maria), HasParent(Maria,Antonio), HasSibling(Leonardo,Vito), HasSibling(Martina,Valentina), HasSibling(Giovanna,Maria), HasSibling(Vito,Leonardo), HasSibling(Tiziana,Claudia), HasSibling(Valentina,Martina), HasChild(Leonardo,Tiziana), HasChild(Antonio,Giovanna), HasChild(Antonio,Maria), HasChild(Giovanna,Tiziana), HasChild(Giovanna,Claudia), HasChild(AntonioB,Leonardo), HasChild(Maria,Valentina), HasUncle(Martina,Giovanna), HasUncle(Valentina,Giovanna) }

From DL to OWL

```
<owl:Class rdf:ID="Human"/>
<owl:Class rdf:ID="Father">
    <owl:equivalentClass>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Class rdf:ID="Man"/>
                <owl:Class rdf:ID="Parent"/>
            </owl:intersectionOf>
        </owl:Class>
    </owl:equivalentClass>
</owl:Class>
<owl:Class rdf:ID="Female">
    <owl:disjointWith>
        <owl:Class rdf:ID="Male"/>
    </owl:disjointWith>
</owl:Class>
<owl:Class rdf:ID="Child">
    <owl:equivalentClass>
        <owl:Restriction>
            <owl:someValuesFrom>
```

Reasoning on ontology

- An ontology is made by a set of axioms $\Rightarrow$ *implicit knowledge* can be *made explicit* (derived) through inferences.
  - Developed Reasoners (based on DL formal semantics):
    - *FaCT*
    - *RACER*
    - *PELLET*

TBox Standard Inference Services

- *Concept Satisfiability:* checks if a newly defined concept makes sense w.r.t. the existing TBox or it is contradictory
  - **Ex.:** $\mathcal{T} = \{$ Parent, Man, Woman $\equiv \neg$ Man, Mother $\equiv$ Woman $\sqcap$ Parent$\}$ *added* Man $\sqsupseteq$ Mother $\Rightarrow$ the new axiom is *unsatisfiable* w.r.t TBox since the disjointness constraint between Man and Woman is violated
- *Subsuption:* checks if a concept C is more general that another concept D $\Rightarrow$ used for computing *concept hierarchy*
  - **Ex.:** $\mathcal{T} = \{$ Parent, Man, Woman $\equiv \neg$ Man, Mother $\equiv$ Woman $\sqcap$ Parent$\}$ *added Father* $\equiv$ *Man* $\sqcap$ *Parent* $\Rightarrow$ Parent $\sqsupseteq$ Father and Man $\sqsupseteq$ Father

ABox Standard Inference Services

- *ABox consistency (w.r.t. the TBox):* checks if a new (concept or role) assertion in an ABox $\mathcal{A}$ makes $\mathcal{A}$ inconsistent w.r.t. a TBox $\mathcal{T}$ or not
  - **Ex.1:** $\mathcal{T} = \{$Woman $\equiv$ Person $\sqcap$ Female, Man $\equiv$ Person $\sqcap \neg$Female$\}$; $\mathcal{A} = \{$ Woman(MARY),Man(MARY)$\} \Rightarrow \mathcal{A}$ is *inconsistent* w.r.t. $\mathcal{T}$
  - **Ex.2:** TBox $\mathcal{T} = \{$Woman, Man$\} \Rightarrow \mathcal{A}$ is *consistent* w.r.t. $\mathcal{T}$ since no restrictions are imposed on the interpretation of Woman and Man
- *Instance Checking:* decide whether an individual is an instance of a concept or not
- *Retrieval:* finds all individuals instance of a concept
  - **Ex.:** $\mathcal{T} = \{$Female $\sqsupseteq$ Woman $\}$; $\mathcal{A} = \{$ Female(Ann), Woman(Sara)$\} \Rightarrow$ *Retrieval*(Female) $= \{$Ann, Sara$\}$

Reasoning Services in Ontology Life-Cycle

Reasoning services can be employed in different phases of the ontology life-cycle

- **Ontology design**
  - Check concept satisfiability, ontology satisfiability and (unexpected) implied relationships
- **Ontology aligning and merging**
  - Assert inter-ontology relationships
  - Reasoner computes integrated concept hierarchy/consistency
- **Ontology deployment**
  - Determine if a set of facts are consistent w.r.t. ontology
  - Determine if individuals are instances of ontology concepts
  - Classification-based querying

Expressiveness and Computability

With the increasing of the expressive power of the knowledge representation language the computational complexity of the reasoning procedure increases ⇒ increasing of the time necessary for computing inferences

- It can happen that some inferences do not give any reply ⇒ semi-decidable procedure

If a conclusion C is not a logic consequence of a set of premises P then the procedure for its prove cannot terminate.

OWL Languages

The OWL language provides three increasingly expressive sublanguages:

- **OWL Lite:** *decidible* with desirable computational properties
  - supports the classification hierarchy inference and simple constraint features, i.e. cardinality constraints on properties where only cardinality values of 0 and 1 are permitted.
- **OWL DL:** *decidible* but subject to higher worst-case complexity. It is so called for its correspondence with DL.
  - allows restrictions such as type separation (a class cannot also be an individual or a property, a property cannot also be an individual or a class).
- **OWL Full:** *not decidible*
  - a class can be treated simultaneously as a collection of individuals and as an individual in its own right.

OWA vs. CWA

- **Open World Assumption:** typical of DL
  - Absence of information is interpreted as unknown information
- **Closed World Assumption:** typical of DB
  - Absence of information is interpreted as negative information

Ex.: $\mathcal{T} = \{$ Female,Woman$\}$;
$\mathcal{A} = \{$ Female(Ann),Woman(Sara)$\}$

CWA: q = Female(Sara) ? $\rightarrow$ *NO*
OWA: q = Female(Sara) ? $\rightarrow$ *UNKNOWN*

Tools for building and managing ontologies

- **Protégé:** free, open source ontology editor and knowledge-base framework
- **Chimaera:** system for creating and maintaining distributed ontologies on the web. Major supported functions: merging multiple ontologies; diagnosing of one or more ontologies.
- **Ontolingua:** distributed collaborative environment to browse, create, edit, modify, and use ontologies.
- **OntoEdit:** Engineering Environment for the development and maintenance of ontologies using graphical means.
- **WebOnto:** Java applet coupled with a customised web server allowing to browse and edit knowledge models over the web.
- **KAON:** open-source infrastructure for ontology creation and management, and providing a framework for building ontology-based applications.

Conclusions

Summarizing

- An *ontology* is a formal conceptualization of a domain that is shared and reused across domains, tasks and group of people
- Ontologies are *used* in different fields *for*:
  - Constituting a community reference
  - Sharing consistent understanding of what information means
  - Making possible interoperability between systems
  - Making the *Web* machine-readable and processable besides of human-readable (Semantic Web)

Line of research:

- *Ontology construction* is the *result of a complex process* of knowledge acquisition $\Rightarrow$ (semi)-automatic tools for building ontologies are necessary
  - *Machine learning methods* can be useful for accomplishing such a goal

The End

That's all!

Claudia d'Amato
`claudia.damato@di.uniba.it`
Nicola Fanizzi
`fanizzi@di.uniba.it`

Bibliography

- N. Guarino and P. Giaretta. Ontologies and Knowledge Bases: Towards a Terminological Clarification. In Proc. of 2nd Int. Conf. on Building and Sharing Very Large-Scale Knowledge Bases (1995).
- A. Gomez Perez and V.R. Benjamins. Overview of knowledge sharing and reuse components: Ontologies and problem-solving methods. In Proc. of Ontology and Problem-Solving Methods: Lesson learned and Future Trends, Workshop at IJCAI, vol. 18, pages 1.11.15. CEUR Pubblications, Amsterdam, 1999.
- *http://en.wikipedia.org/wiki/Ontology_(computer_science)*
- *http://www.w3.org/2004/OWL/*

Bibliography

- John F. Sowa, Knowledge Representation: Logical, Philosophical, and Computational Foundations, Brooks Cole Publishing Co., Pacific Grove, CA, 2000 *http://www.jfsowa.com/krbook/*
- S. Staab, R. Struder (Eds.), Handbook on Ontolgies, Springer
- A. Gómez-Pérez. Ontological Engineering. Tutorial at IJCAI99 *http://www.ontology.org*
- Protégé *http://protege.stanford.edu/*