Chapter 1

THE ROAD TOWARD ONTOLOGIES

Diana Marcela Sánchez, José María Cavero and Esperanza Marcos Martínez
Universidad Rey Juan Carlos Departamento de Informática, Estadística y Telemática

Abstract: One of the most important characteristics of today's society is that a huge amount of information is shared by many participants (people, applications). This information must be characterized by a uniformity of terms. This means that, in similar contexts, everyone should understand the same meaning when reading or hearing the same word and everyone should use the same word to refer to the same concept. In different Computer Science disciplines one of the methods that satisfies this need for "common understanding" of concepts is the creation of *ontologies*. Curiously, there are different interpretations of what ontology is. In this chapter, we show the way that the concept of ontology has expanded from Philosophy into Computer Science.

Key words: Ontology; Philosophy; Computer Science

1. INTRODUCTION

As in many other disciplines, in Computer Science new terms emerge and become fashionable. In recent years one of these terms is the concept of *ontology*. It has been adopted by Computer Science with a different meaning than it had in its origin. Ontology in Computer Science, broadly speaking, is a way of representing a common understanding of a domain.

Perhaps one of the consequences of the World Wide Web is the idea that all of the world's knowledge is available to everyone. Although this is obviously not correct, it has created new demands on Computer Science. The idea of sharing knowledge requires that all participants (not only people, but also applications) must share a common vocabulary, that is, a consensus about the meaning of things. Ontologies, therefore, are one of the solutions for representing this common understanding. However, the concept of ontology had a long history in Philosophy before being used in Computer Science.

The rest of this chapter is organized as follows. In section two, we summarize the traditional (philosophical) definition of ontology. Section three presents how the *ontology* concept came to be used in Computer Science and later reviews the meaning of this concept in that discipline, including different classifications of ontologies within Computer Science. Section four shows how ontologies are used in the development of Information Systems, including techniques and methodologies for developing ontologies, and applications based on ontologies. Finally, Section five offers conclusions and suggests possible future work.

2. PHILOSOPHICAL ONTOLOGY

The concept of ontology was taken from Philosophy and applied to Computer Science with a meaning different from the one traditionally accepted since Classical Greece. In the following paragraphs, we take a look at the classical definition of ontology and related concepts, starting with Aristotle. We conclude by discussing the adoption of the concept by some of the Computer Science disciplines.

Since early human history, people have asked themselves about the "essence" of things. Aristotle, in his *Metaphysics*, was one of the first philosophers to ask and to write about "*What is being?*" In an attempt to answer that question, he concluded that all beings in the world must have some "thing", some characteristic, which gives the property of "being" to objects.

Aristotle starts his Metaphysics with a compilation of the different approaches to the meaning of the primary constitutive element (the essence of things) and how that essence or primary element generates all things in the world. For example, Anaximenes thought that air was the first principle of all things, and Tales thought that the water was the beginning, reason and substance of all things. Aristotle, nevertheless, thought that those approaches dealt with the primary principle rather than the essence of things.

He distinguished between principle and essence. Principle is the "source point of something" while essence is the "intrinsic reason of existence of being" (Aristotle, 1994).

Nevertheless, neither Ontology nor Metaphysics were concepts used by Aristotle in his essays. Andronicus of Rhodes, who divulged Aristotle's writings for the first time, was the one who observed that the main subject of Aristotle's writings went *beyond* Physics. It was he who coined the term Metaphysics. In the middle ages, metaphysics studies were influenced by the idea of God. God is presented as *the Creator* of all things, a divine, transcendent and mystic being capable of giving life, that is, of giving "essence". But God is a particular being; therefore, the study of God (Theology) could not replace Metaphysics in its search for the intrinsic reason common to *all* beings. Philosophers of the Modern Ages applied the "divide and conquer" strategy, so they divided the study of beings according to the nature of objects studied. Nevertheless, their discussions and conclusions were still grouped around Metaphysics.

By the end of XVII century, Christian Wolff divided Metaphysics into "*metaphysica generalis*" and "*metaphysica specialis*". *Metaphysica generalis* (general metaphysics) was also called "*ontologia*" (Ontology), with the meaning to investigate the most general concepts of being. Meanwhile "*metaphysica specialis*" (special metaphysics) was divided into three branches: Rational Theology (the study of God), Rational Psychology (the study of the soul) and Rational Cosmology (the study of the universe as a whole) (García Sierra, 1999).

Traditionally, philosophers have adopted two attitudes about General Metaphysics (that is, Ontology). Both look for the "essence" of things, but with different approaches:

1. The first approach looks for the intrinsic reason that might allow us to give the name "being" to objects that possess it. The method to obtain that *essence* must be through observation of and reflection on all things and behaviors in the world, and then put such reasoning into words.
2. The second approach also looks for essence, but through a hierarchical classification of beings. In this classification, high levels are generated by general properties and could be composed by lower levels, which represent more specific characteristics. An organization of beings permits us to find common characteristics (physical or not). The top level of this classification must be the essence, that is, the property that all beings (animated or not) possess and permit them to exist.

3. FROM PHILOSOPHY TO COMPUTER SCIENCE

It could be thought that ontologies entered Computer Science through Philosophy of Science, which is a branch of Philosophy that looks for the reason and justification of sciences (Mosterin, 2000). Nevertheless, the path followed by the Ontology concept from Philosophy to Computer Science was the result of different requirements in various fields (Smith and Welty, 2002). Artificial Intelligence, Software Engineering and Database communities independently concluded that knowledge representation was important for the evolution of their areas.

In the field of Artificial Intelligence (AI), this need for knowledge representation is most evident because its goal is to make an agent do tasks autonomously and systematically (Guarino and Giaretta, 1995). To do a job well, agents must make decisions; these decisions must be made based on knowledge. So, AI's researchers' goal is to incorporate knowledge into agents, that is, to find a method of representing knowledge in a computational environment. Epistemology, which is "the field of Philosophy which deals with the nature and sources of knowledge" (Nutter, 1987) can help to find answers and tools to create a way of representing knowledge. Regarding Epistemology, if we speak about the nature of knowledge, then we ask ourselves about its components. If we speak about the sources of knowledge, we try to understand the inference process we use to generate it. One of the most widely accepted ideas in the epistemological field is that knowledge is made up of concepts.

The object-oriented paradigm gave Software Engineering a new style of representing elements involved in a problem. This paradigm classifies the world into objects that may be the representation of anything of the world. Those elements have two basic characteristics: attributes (or properties) and methods (or possible actions that objects could do) (Booch, 1993). Object-orientation is a hierarchical way of thinking about the world where an object inherits properties and methods from its parents. Objects may have different behaviors depending on the situation, due to the *overloading* mechanism, which allows giving multiple meanings to the same concept. Polymorphism is a property of the objects that allows them to answer a specific requirement in different ways, according to their particular properties. At a higher level, software engineers found that representing concepts, that is, representing the meaning of things, may also help to simplify some problems, like systems interoperability.

Finally, the Database community also needed conceptual, high level models. The purpose of such models was to give an abstract representation of a problem domain without considering implementation issues.

Therefore, three different areas had the same problem: the representation of concepts. This representation can be used, for example, as a starting point to generate knowledge.

According to the Oxford Dictionary (Oxford, 1993) a concept is an abstract idea and has a Latin origin which means "something conceived". So, a concept is the representation of the meaning of a thing or, in other words, the mental representation of an object when a human being thinks about that object. Concepts take on the main characteristics of things, that is, their essence.

However, each Computer Science discipline addressees the problem of knowledge representation in a different manner, because each one is interested in a specific problem, called problem domain (Guarino, 1998). Therefore, researchers elaborate a valid representation for a specific part of reality.

In 1980, John McCarthy proposed, in the field of Artificial Intelligence, the concept of an environment's ontology (McCarthy, 1980). An environment's ontology comprises not only a list of concepts involved in a problem (environment) but also their meanings in that context, that is, what we mean by each one of them. He applied ontologies for establishing an order in the concepts within a domain. Since then, ontologies have been associated with the representation of concepts.

Before continuing, it is important to take into account two issues:

1. We are talking about concepts. Therefore, we have to think about a *conceptualization* process, that is, the process by which concepts are generated.
2. We are talking about representation. Therefore, we want to "present" something; in other words, we want to express the characteristics and structure of things easily (McCarthy, 1980). In Computer Science, the mechanism used to show the structure (to present) has always been the creation of models.

Doing a rough comparison with philosophical Ontology, one might come to some preliminary conclusions: Computer Science does not give an answer about what is the essence (it is not its goal). It assumes that everything that can be represented is "real". In this context, concepts are primary principles, so all the things that exist in the world are susceptible to being represented by a concept which tries to capture its meaning (essence). Computer Science models are constructed for small, reduced domains; if those models were hierarchical, then when modeling, we were looking for the primary elements of our reduced domain. That is the same goal that philosophical ontology has for the entire world.

Currently, the most common definition of ontology in Computer Science is Gruber's (Gruber, 1993): ontology is an "explicit specification of a conceptualization". This definition is based on the idea of conceptualization: a simplified view of the world that we want to represent. Conceptualization is the process by which the human mind forms its idea about part of the reality. This idea is a mental representation free of accidental properties and based on essential characteristics of the elements. Therefore, the (Computer Science) ontology concept is joined to a domain or mini-world and the specification represented in ontology is concerned with that domain. In other words, if the domain (or part of it) changes, the conceptualization must also change and consequently the ontology that represents this mini-world changes too.

Regards the Gruber definition, a lot of comments and new definitions has been proposed by several authors– within Computer Sciences disciplines-. All these definitions are based on the idea that Computer Science ontology is a way of representing concepts.

Some authors have compared philosophical and Computer Science ontology concepts. Guarino proposes that both concepts be distinguished using different terms. He proposes "ontology" as the Computer Science term and "conceptualization" for the philosophical idea of "*search for the essence of beings*" concept (Guarino and Giaretta, 1995). He argues that currently the Ontology concept has taken on a concrete meaning and that it is associated with the development of a model which often represents a particular situation. He observes that the term "conceptualization" should be used for the abstract and non palpable process of reasoning, and "ontology" for the concrete process of reasoning. Nevertheless, as we have previously said, that "process for the creation of concepts" belongs to Epistemology (the way how knowledge is generated) more than to philosophical Ontology.

The next step in the construction of ontologies is to explicitly represent conceptualization, that is, select which tool may be used to represent knowledge. One attempt to formally represent conceptualizations is the concept of Formal Ontology. It looks, using Logics, like "an axiomatic and systematic study about all forms of being" (Cocchiarella, 1991).

Formal Ontology is, for several authors, the Theory of Distinctions at all levels (Guarino, 1998). Theory of Distinctions may be applied to entities or to categories of entities, or even to categories of categories (meta-categories) that used the world for modeling. For other authors, it is the study of formal structures to represent knowledge and its relations. However, for both approaches, there are two important study fields associated with Formal Ontology: Mereology, or the study of part-whole relations, and Topology, or study of connection relationships (Guarino, 1998; Smith, 1998).

The purpose of Mereology is to identify when an object (that may be composed of other objects) stops being itself and turns into something else (by aggregating a new component or subtracting one of its components) (Mosterin, 2000). This is very important in Computer Science's ontologies, because in a hierarchical model, where a concept is divided into its components, it is important to distinguish where the limit between essential and non essential elements is. Essential elements are those components of the element that if they disappear, the (composed) element changes or no longer exists. At this point, Topology can help. Topology analyzes the strength of the relationships between elements. Using Topology, we can compare two elements and decide if they are the same element; or if an element is essential for another element, which means that they can not "live" separately.

Those previous concepts may be applied to interoperability between systems. If we were able to know what the essential characteristics that distinguish an object are, then it might be possible to analyze another information system and find the element that possesses the same characteristics.

We have previously said that the purpose of ontologies is to represent concepts. But, how do those concepts end up being "real" in some (human or artificial) system? We could say that any representation needs a language to be expressed; ontologies are no exception.

Formal ontology distills, filters, codifies and organizes the results of an ontological study (in either it's local or global settings). (Poli, 2004). So, in Computer Science, Formal Ontology represents ontologies through logical structures. A formalization of ontology is given in a logical language, which describes a structure of the world that considers all objects involved within the domain of study, their possible states and all relevant relationships between them.

3.1 Classification of Ontologies

There are several classifications of Computer Science's ontologies, based on different parameters. Guarino (1998) classifies them by their level of generality in:

- top-level ontologies, which describe domain-independent concepts such as space, time, etc., and which are independent of specific problems;
- domain and task ontologies which describe, respectively, the vocabulary related to a generic domain and a generic task;
- and, finally, application ontologies, which describe concepts depending on a particular domain and task.

Van Heijst, Schereiber and Wieringa (1996) classify them according to their use in:

- terminological ontologies, which specify which terms are used to represent the knowledge;
- information ontologies, which specify storage structure data; and
- knowledge modeling ontologies, which specify the conceptualization of the knowledge.

Fensel, (2004) classifies ontologies in:

- domain ontologies, which capture the knowledge valid for a particular domain;
- metadata ontologies, which provide a vocabulary for describing the content of on-line information sources;

- generic or common sense ontologies, which capture general knowledge about the world providing basic notions and concepts for things like time, space, state, event, etc;
- representational ontologies, that define the basic concepts for the representation of knowledge; and
- finally, method and particular tasks ontologies, which provide terms specific for particular tasks and methods. They provide a reasoning point of view on domain knowledge.

Gómez-Perez, Fernández-López and Corcho (2003) classify ontology based on the level of specification of relationships among the terms gathered on the ontology, in:

- Lightweight ontologies, which include concepts, concept taxonomies, relationships between concepts and properties that describe concepts.
- Heavyweight ontologies which add axioms and constraints to lightweight ontologies. Those axioms and constraints clarify the intended meaning of the terms involved into the ontology.

4. WORKING AROUND ONTOLOGIES

Since their appearance, ontologies have been one of the most important branches of development in Computer Science. As in any new area of knowledge, when researchers started to work with ontologies almost everything had still to be done. However, the needs of researchers in this area focused on three specific activities: Techniques, Methodologies and Applications. All of these activities could be compiled under Ontological Engineering. According to Gómez-Perez, Fernández-López and Corcho (2003), Ontological Engineering refers to the set of activities that concerns the ontology development process, the ontology life cycle, and the methodologies, tools and languages for building ontologies.

In the following sections, we briefly summarize some techniques, methodologies and applications related to ontologies, with the aim of giving a general outlook about work in this field of study.

4.1 Techniques

Any formalism used to materialize ontology must contain elements for representing concepts and their relations. Those elements are always based on a set of basic axioms that set the parameters and the representation rules.

Some initiatives for the modeling of ontologies are:

- (Gruber, 1993) proposes using frames and first order logic. This schema uses classes, relations, functions, formal axioms and instances. Classes are the representation of relevant concepts (no matter if they are abstract or specific concepts) in the domain; classes are organized in taxonomies. Relations represent different types of associations between concepts in a domain. Functions are a special case of relations. Other elements are the formal axioms, which are sentences that are always true; these axioms are used to generate new knowledge and to verify consistency of the ontology. Finally, instances are used to represent elements or individuals in the ontology.

- Another proposal for modeling ontologies is using Description Logics (DL) (Baader, Horrocks and Sattler, 2004). DL is a logical formalism that is divided in two branches: TBox and ABox. The TBox contains the definitions of concepts and roles, also called intentional knowledge; the ABox contains the definitions of individuals, also called extensional knowledge. Therefore, systems based on DL use three elements to represent ontologies' components: concepts, roles and individuals. Concepts represent classes of objects, roles describe relations between concepts, and individuals represent instances of classes. Concepts and roles are specified based on a set of pre-existing terms and constructors whose elements can be mixed to obtain any kind of DL language. Primitive concepts are those whose specification does not need to be based on other concepts, but only on conditions that individuals must satisfy. Derived concepts are those concepts whose specification is based on another concept, from which it inherits some properties. Individuals represent an instance of the concepts and their values.
- Software Engineering Techniques like Unified Modeling Language (UML) are also used for modeling ontologies. Several authors argue that basic UML is enough to represent lightweight ontologies (Cranefield and Purvis, 1999 ; Kogut et al., 2002), however, for heavyweight ontologies it is necessary to enrich UML with, for example, the Object Constraint Language (OCL). OCL is the language for describing constraints in UML and helps us to formalize its semantics. UML class diagrams are the diagrams used to represent concepts where each class represents a concept. The instances of classes are represented by objects, which are instances of a concept. Concept taxonomies are represented through generalization relationships. Binary relations are represented through association relationships.
- Database Technologies are another possibility to represent ontologies (Gómez-Perez, Fernández-López and Corcho, 2003) using, for example, Entity-Relationship (ER) diagrams. In these diagrams, concepts can be represented using entities, which have attributes that are the properties of the concept. These attributes have a name and a type. Relations between concepts are represented by relationships, which have cardinality and

permit expression not only of associations but also generalization relations to create taxonomies of the concepts. Formal axioms can be represented using integrity constraints.

4.2 Methodologies

Like any piece of software, the construction of ontologies may be improved if some kind of methodology is applied. The goal of using a methodology is to obtain a good result following a set of steps which usually are based on best practices.

Most of the methodologies for building ontologies are based on the experience of people involved in their construction. In several cases, methodologies are extracted from the way in which a particular ontology was built.

Nowadays, methodologies are more focused on modeling knowledge than on developing applications. So, such methodologies are good alternatives for modeling knowledge instead of good alternatives for managing an information technology project centered on ontologies.

Next, we briefly summarize some significant methodologies that can be found in the literature. First we are going to list methodologies designed to build ontologies. The steps that confirm the methodologies are the result of analyzing the good choices and the mistakes in projects formulated to create ontology for a particular case:

- *Cyc* is based on the experience during the development of the Cyc knowledge base (Lenat and Guha, 1990), which contains a great quantity of common sense knowledge. In this process three basic tasks were carried out:
  - First, manual extraction of common sense knowledge;
  - Second, the knowledge coding was aided by tools using the knowledge already stored in the Cyc knowledge base;
  - Third, computer managed extraction from common sense knowledge.

  CycL was the language used to implement Cyc and two activities were carried out to specify the ontology:
  - First activity: development of a knowledge representation and a top level ontology with the most abstract concepts, and
  - Second activity: representation of the knowledge for different domains
- (Uschold and King, 1995) is one of the first specific proposals for building ontologies. It was used for developing *Enterprise Ontology* and for describing a set of guidelines to create an ontology:
  - To identify the purpose of the ontology.
  - To build the ontology through three activities. The first activity consists of capturing the ontology, in which we capture the concepts and the

relationships between concepts. The second activity consists of codifying the ontology using a formal language. The third activity consists of integrating the resulting ontology with previously existing ones.

– To evaluate the ontology, that is, "to make a technical judgment of the ontology, their associated software environment, and documentation with respect to a frame of reference".

- (Grüninger and Fox, 1995) methodology was developed to implement Toronto Virtual Enterprise (TOVE) ontology and is divided into six steps:

– To identify motivating scenarios: to capture the why? And what for? for which the ontology is built

– To elaborate informal competency questions: this consists of asking some questions written in natural language that must be answered by the ontology. These questions will be used to delimit the restrictions of the ontology and to evaluate the final ontology.

– To specify the terminology using first order logic.

– To write competency questions in a formal way using formal terminology: the questions used in step 2, are re-written in first order logic.

– To specify axioms using first order logics: this methodology proposes using axioms to specify the definitions of terms in the ontology and constraints in their interpretation.

– To specify completeness theorems: To define several conditions to assure that the ontology is finished.

- *Amaya* methodology is the result of ESPRIT KACTUS project (KACTUS, 1996), which investigates the possibility of reusing knowledge in complex technical processes. The method has three stages:

– To specify the application, where we identify context and the elements that we want to model.

– Preliminary design based on relevant top-level ontological categories. The elements identified in the previous step are used as inputs to obtain a global vision of the model. During this process it is possible to establish the reuse of ontology that already exist.

– Ontology refinement and structuring. To specialize terms of ontology for obtaining a definitive design with the maximum modularization.

The following methodologies address the development of ontologies in the framework of software projects. Therefore, their purpose is more focused on developing software applications which main elements are ontologies.

- *CommonKADS*: (Schreiber et al., 1999). Although it is not a methodology, it covers several aspects from corporate knowledge management to implementation of knowledge information systems.

CommonKADS has a focus on the initial phases for developing knowledge management applications.

- *Methontology* (Gómez-Perez, Fernández-López and Corcho, 2003) is inspired by software development methodologies. This methodology divides the process into three phases:

1. Project management activities. Those activities involve the planning, tracking of task and control of quality to obtain a good result.
2. Development-oriented activities: Specification of the ontology, formalization of resources used to build the ontology, design, implementation and maintenance.
3. Support activities: Knowledge gathering, ontology evaluation, ontology reuse and documentation.

This methodology divides the process for modeling knowledge process into eight tasks:

– Task 1: To build the glossary of terms. Those terms must have their natural language definition, their synonyms and their acronyms.
– Task 2: To build concept taxonomies to classify the concepts.
– Task 3: To build ad hoc binary relation diagrams to identify ad hoc relationships between concepts of the ontology or concepts of other ontologies.
– Task 4: To build a concept dictionary. A concept dictionary contains all the domain concepts, their relations, their instances and their classes and instance attributes.
– Task 5: To describe in detail each *ad hoc* binary relation that appears in the binary relation diagram. Results of this task are shown in an *ad hoc binary relation table.*
– Task 6: To describe in detail each instance attribute that appears on the concept dictionary.
– Task 7: To describe in detail each class attribute that appears on the concept dictionary.
– Task 8: To describe each constant, which specifies information related to the knowledge domain.

4.3 Applications

Several branches of Computer Science have used ontologies to model their knowledge. Database Systems, Software Engineering and Artificial Intelligence are the three most important fields where ontologies have been used to construct solutions to satisfy their needs.

The main purpose for using ontologies in previous branches of Computer Science is as a means of integrating several platforms or applications. The problem of integration between platforms consists of looking for the most natural way to inter-communicate two applications. To obtain such communication, it is important to have a set of concepts that compile vocabulary used by the applications and a set of rules for solving semantic heterogeneity that could exist between the concepts in each application. The association of these two elements allows transforming data from one application to another. So, the solution developed must allow information sharing and have efficient communication (Rubin, 2003), (Zhibin, Xiaoyong, Ishii, 1998), (Dehoney, Harte, Lu, Chin, 2003), (Tosic and Agha, 2004).

Another common use of ontologies is for domain modeling. Ontologies constitute a good alternative for representing the shared knowledge about a domain. Leaving aside accidental characteristics, ontologies hope to represent an objective point of view of a part of the reality, so its representation is more universal and includes the main characteristics that would be used by any application that is expected to give a particular solution in a modeled domain (Wagner and Taveter, 2004; Dehoney, Harte, Lu, Chin, 2003; Sallantin, Divol, Duroux, 2003).

It is also possible to apply ontologies to support specific tasks in different fields of study. In Database Systems, the ontologies help to model a specific domain and facilitate the integration with other databases. In addition, they improve information search (Kohler, Lange, Hofestadt, Schulze-Kremer, 2000). In Software Engineering, a specific ontology could be taken as reference point to validate a model that acts over a particular domain (Ambrosio, de Santos, de Lucena, da Silva, 2004; Conesa, de Palol, Olivé, 2004), likewise several paradigms of Software Engineering, like, for example, Extreme Programming could be used to build ontologies (Ceravolo, Damiani, Marchesi, Pinna, Zavaterelli, 2003). In Artificial Intelligence, ontologies help to ease the inference process (Rubin, 2003).

Figure 1-1 shows, by means of a use case diagram, the different ways ontologies could be used in Software Engineering, Artificial Intelligence and Database Systems.

So, ontology is found in a wide range of applications and may take different forms. In the following, some application examples of different topics are briefly summarized:

- FLAME 2008 is a platform to model a provider system of services for mobile devices based on ontologies (Weißenberg, Voisard and Gartmann, 2004).
- ONTOLOGER is an application for optimizing the searching on the Web according to user profiles (Stojanovic, González and Stojanovic, 2003).
- Carr et al. (2001) create a conceptual hypermedia service which provides hyperlinks for searching on the web. Hyperlinks are obtained by an improved ontological processing.

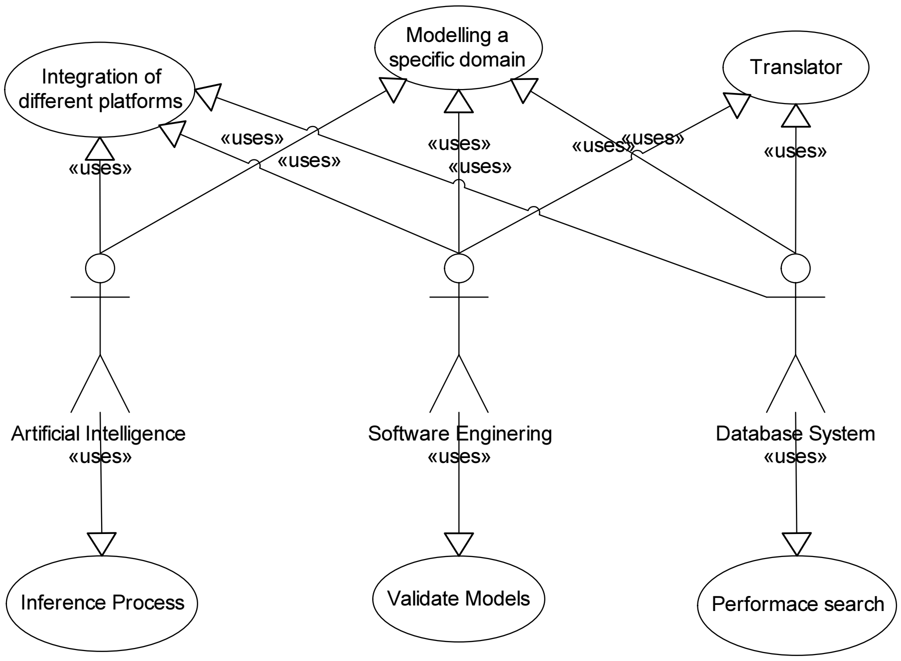

Figure 1-1. [Uses of Ontologies]

- *OntoWeb Project* is a thematic network created to exchange information in fields like knowledge management or e-commerce using ontologies. (Oberle and Spyns, 2004).
- *Onto-Share* is an application for virtual communities where ontologies specify a hierarchy of concepts (Davies, Duke and Sure, 2003).
- *SweetDeal* is a rule-based approach based on an ontology for the representation of business contracts (Grosof and Poon, 2003).
- Corcho et al. (2003) present a three-layer ontology-based mediation framework for electronic commerce applications.
- *CoursewareWatchdog* use ontologies to model an easy e-learning system (Tane, Schmitz and Stumme, 2004).
- *Edutella* is a P2P network based on the use of an ontology (KAON) for the exchange of educational resources between German universities (Nejdl et al., 2001).
- *OntoVote* is a mechanism for the development of ontologies in P2P environments, so ontology is produced by the consensus of all members of P2P application (Ge et al., 2003).
- SAMOVAR (Systems Analysis of Modeling and Validation of Renault Automobiles) is a system developed based on ontologies to optimize the

design of automobiles through management of past design experiences (Golebiowska et al., 2001).
- Stevens et al. (2004) show us a general panorama of using ontologies in bioinformatics which is a discipline that uses computational and mathematical techniques to manage biological data.
- Thesaurus and tools that organizes knowledge in concepts and relations, such as The Art and Architecture Thesaurus (AAT, 2005), WordNet (WordNet, 2004), ACM Computing classification system (ACM, 1998).

A survey about the relationship between ontologies and information systems can be found on (Chandrasekaran, Josephson, Benjamins, 2003)

5. CONCLUSIONS AND FUTURE WORK

The concept of *ontology* has been taken up by Computer Science with a different meaning than the one that it traditionally has had in Philosophy for centuries. In this work we have summarized the evolution of the concept of ontology as it passed from Philosophy to Computer Science, and we have examined the new meaning that this term has acquired. In future works, a profound comparison of this concept with related Computer Science terms (for example, the concept of *model*) will be addressed.

ACKNOWLEDGEMENTS

This research was carried out in the framework of the MIFISIS project (*Research Methods and Philosophical Foundations in Software Engineering and Information Systems*) supported by the Spanish Ministry of Science and Technology (TIC2002-12378-E), and of the PPR-2004-15 project, financed by Rey Juan Carlos University.

REFERENCES

AAT, 2005, Art & Architecture Thesaurus Online; (March 7, 2005) http://www.getty.edu/research/conducting_research/vocabularies/aat/.

ACM, 1998, The ACM Computing Classification System, 1998 Version. http://www.acm.org/class.

Ambrosio, A.P., de Santos D.C., de Lucena F.N., da Silva J.C., 2004, Software engineering documentation: an ontology-based approach, *WebMedia and LA-Web, 2004. Proceedings*. pp. 38 – 40.

Aristotle, 1994, *Metaphysics*. Chapters I and V. Oxford: Oxford University Press. Oxford.

Baader, F., Horrocks, I., Sattler, U., 2004, Description Logics. In: Staab S and Studer R, ed. *Handbook on Ontologies*. Berlin: Springer-Verlag, pp. 03-28.

Booch, G., 1993. *Object-oriented Analysis and Design with Applications* (2nd edition). Canada: Addison-Wesley Press.

Carr, L., Hall, W., Bechofer, S., Goble, C., 2001, Conceptual Linking: Ontology-Based Open Hypermedia. *International World Wide Web Conference 2004,* pp. 334-342. New York: ACM Press.

Chandrasekaran, B., Josephson, J.R., Benjamins, V.R., 2003, What are ontologies, and why do we need them? *Intelligent Systems and Their Applications, IEEE. IEEE Intelligent Systems* Volume 14, Issue 1. pp. 20 – 26.

Cocchiarella, N.B., 1991, Formal Ontology. In: Burkhard, H. and Smith, B. (eds), *Handbook of Metaphysics and Ontology*. Munich: Philosophia Verlag.

Conesa, J., Palol, X. de, Olivé, A., 2003, Building Conceptual Schemas by Refining General Ontologies. In *Marik, V, et al (ed). DEXA 2003*. pp: 693-702.

Corcho, O., Gómez-Pérez, A., Leger, A., Rey, C., Toumani, F., 2003, An Ontology-based Mediation Architecture for E-commerce Applications. *IIS 2003*; pp. 477-486.

Cranefield, S., Purvis, M., 1999, UML as an Ontology Modelling Language. In: Fensel D, Knoblock C, Kushmeric N and Rousset MC, ed. *IJCAI'99 Workshop on Intelligent information integration. Stockholm, Sweden.* Amsterdam, 5.1-5.8.

Davies, J., Duke, A., Sure, Y., 2003, OntoShare - A Knowledge Management Environmental for Virtual Communities of Practice. *International Conference On Knowledge Capture Sanibel Island 2003*. New York. ACM Press

Dehoney, D., Harte, R., Lu, Y., Chin, D., 2003, Using natural language processing and the gene ontology to populate a structured pathway database. *Bioinformatics Conference, 2003. CSB 2003. Proceedings of the 2003 IEEE*. pp. 646 – 647.

Fensel, D., 2004, *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*. 2nd ed. Berlin: Springer-Verlag

García Sierra, P., 1999, *Diccionario filosófico. Manual de materialismo filosófico. Una introducción analítica.* Oviedo: Biblioteca Filosofía en español.

Ge, Y., Yu, Y., Zhu, X., Huang, S., Xu, M., 2003, OntoVote: A Scalable Distributed Vote-Collecting Mechanism For Ontology Drift On A P2P Platform, *The Knowledge Engineering Review,* 18(30): 257-263. New York: Cambridge University Press.

Golebiowska, J., Dieng-Kuntz, R., Corby, O., Mousseau, D., 2001, Building And Exploiting Ontologies For An Automobile Project Memory, *International Conference On Knowledge Capture,* pp. 52-59. New York: ACM Press

Gómez-Pérez, A., Fernández-López, M., Corcho, O., 2003, *Ontological Enginering*. London, Springer-Verlag.

Grosof, B., Poon, T., 2003, Sweetdeal: Representing Agent Contracts With Exceptions Using XML Rules, Ontologies And Process Descriptions, *International World Wide Web Conference.* pp: 340-349. New York: ACM Press.

Gruber, T.R., 1993, A translation approach to portable ontology specifications. *Knowledge Acquisition* 5(2): 199-220

Grüninger, M., Fox, M.S., 1995, Methodology for the design and evaluation of ontologies, In: Skuce, D., ed. *IJCAI 95 Workshop on basic ontological issues in knowledge sharing*., 6.1-6.10.

Guarino, N., Giaretta, P., 1995, Ontologies and Knowledge Bases: Towards a Terminological Clarification. In: NJI Mars, ed. *Towards very large knowledge bases*. Amsterdam: IOS Press, pp: 25-32

Guarino, N., 1998, Formal Ontology and Information Systems. In: *FOIS'98*, Trento, Italy. Amsterdam: IOS Press. pp. 3-15

KACTUS, 1996, The KACTUS Booklet version 1.0 Esprit Project 8145 KACTUS, (April 5, 2005); http://www.swi.psy.uva.nl/projects/NewKACTUS/Reports.html.

Kogut, P., Cranefield, S., Hart, L., Dutra, M., Baclawski, K., Kokar, M., Smith, J., 2002, UML for Ontology Development. *The Knowledge Engineering Review,* 17(1); 61-64

Kohler, J., Lange, M., Hofestadt, R., Schulze-Kremer, S. Logical and semantic database integration, 2000, *Bio-Informatics and Biomedical Engineering, 2000. Proceedings. IEEE International Symposium on,* pp. 77 – 80.

Lenat, D.B., Guha, R.V., 1990, Building large knowledge-based systems: Representations and Inference in the Cyc Project. Addison-Wesley, Boston Massachusetts.

McCarthy, J., 1980, Circumscription – A form of non-monotonic reasoning. *Artificial Intelligence*, 13; 27-39

Mosterín, J., 2000 *Conceptos y teorías en la ciencia*. Editorial Alianza. Madrid

Nejdl, W., Wolf, B., Staab, S., Tane, J., 2001, EDUTELLA: Searching and Annotating Resources within an RDF-based P2P Network. (April 27, 2005); http://edutella.jxta.org

Nutter, J.T., 1997, Epistemology. In: S. Shapiro, ed. *Encyclopedia of Artificial Intelligence*. Wyley Press.

Oberle, D., Spyns, P., 2004, The Knowledge Portal "OntoWeb", In: Staab, S. and Studer, R., ed. *Handbook on Ontologies*. Berlin: Springer-Verlag, pp: 499-516.

Oxford, 1993, *Compact Oxford Dictionary*. Oxford University Press

Poli, R., 2004, *Descriptive, Formal and Formalized Ontologies*. University of Trento. Mitteleuropa Foundation

Rubin, S.H., 2003, On the fusion and transference of knowledge. II. *Information Reuse and Integration, 2003. IRI 2003. IEEE International Conference on*. pp. 150 – 159

Sallantin, J., Divol, J., Duroux, P., 2003, Conceptual framework for interactive ontology building, *Cognitive Informatics, 2003. Proceedings. The Second IEEE International Conference on*. pp. 179 – 186.

Schreiber, G., Akkermans, H., Anjewierden, A., Hoog, R. de, Shadbolt, N., Velde, W. van de, Wielinga, B., 1999, Knowledge Engineering And Management – The CommonKDAS Methodology. The MIT Press, Cambridge, Massachusetts.

Smith, B., Welty, C., 2002, Ontology: Towards a New Synthesis. *FOIS Introducction 2002*. Amsterdam: IOS Press

Smith, B., 1998, The Basic Tools of Formal Ontology, In: Nicola Guarino (ed.). *Formal Ontology in Information Systems.* Amsterdam, Oxford, Tokyo, Washington, DC: IOS Press (Frontiers in Artificial Intelligence and Applications), pp. 19-28.

Stevens, R., Wroe, C., Lord, P., Goble, C., 2004, Ontologies In Bioinformatics. In: Staab, S. and Studer, R., ed. *Handbook on Ontologies*. Berlin: Springer-Verlag, pp. 635-657

Stojanovic, N., Gonzalez, J., Stojanovic, L., 2003, ONTOLONGER – A System For Usage-Driven Management Of Ontology-Based Information Portals, *International Conference On Knowledge Capture* Sanibel Island, New York: ACM Press, pp. 172-179

Tane, J., Schmitz, C., Stumme, G., 2004, Semantic Resources Management for the Web: An E-Learning Application, *International World Wide Web Conference,* pp. 01-10. New York: ACM Press.

Tosic, P. T., Agha, G. A., 2004, Towards a hierarchical taxonomy of autonomous agents. *Systems, Man and Cybernetics, 2004 IEEE International Conference on*. 4:3421–3426

Uschold, M., King, M., 1995, Towards a Methodology for building ontologies . In: Skuce D, ed. *IJCAI'95m Workshop on Basic Ontological Issue in Knowledge Sharing*. Montreal, 6.1-6.10.

Van Heijst, G., Schereiber, A. T., Wielinga, B. J., 1996, Using Explicit Ontologies in KBS Development. *International Journal of Human and Computer Studies*.

Wagner, G., Taveter, K., 2004, Towards radical agent-oriented software engineering processes based on AOR modeling. Intelligent *Agent Technology, 2004. (IAT 2004). Proceedings. IEEE/WIC/ACM International Conference on*. pp. 509 – 512.

Weißenberg, N., Voisard, A., Gartmann, R., 2004, Using Ontologies In Personalized Mobile Applications. *The 12th Annual ACM International Workshop on Geographic Information Systems*. New York: ACM Press.

WordNet, 2005, WordNet A lexical database for the English language. 2005, (February 3, 2005); http://wordnet.princeton.edu.

Zhibin, L., Xiaoyong, D., Ishii, N., 1998, Integrating databases in Internet. *Knowledge-Based Intelligent Electronic Systems, Proceedings KES '98. 1998 Second International Conference on*. 3: 21-23.